N° d'ordre : 3694

THÈSE

présentée

devant l'Université de Rennes 1

pour obtenir

le grade de Docteur de l'Université de Rennes 1
Mention Mathématiques et Applications

par

Thomas Sirvent

Institut de Recherche Mathématique de Rennes
École Doctorale Matisse
U.F.R. de Mathématiques

Titre de la thèse :

Courbes elliptiques et applications cryptographiques à la diffusion numérique sécurisée

Soutenue le 26 septembre 2008 devant la commission d'examen

Rapporteurs	M.	Jean-Marc	Couveignes	Université Toulouse 2
	M.	David	Pointcheval	École Normale Supérieure de Paris
Examinateurs	M.	Marc	Girault	Orange Labs de Caen
	M.	Louis	Goubin	Université de Versailles Saint-Quentin
	M.	David	Lubicz	DGA/CÉLAR et Université de Rennes 1
Directeurs	M.	Reynald	Lercier	DGA/CÉLAR et Université de Rennes 1
	M.	Louis	Mahé	Université de Rennes 1

À Émilie,
au bonheur qu'elle me donne,
au soutien qu'elle m'apporte.

Remerciements

Je voudrais tout d'abord remercier Jean-Marc Couveignes et David Pointcheval, qui ont accepté la lourde tâche de rapporter ce manuscrit. Leurs conseils, leurs remarques ont grandement contribué à améliorer ce document. Je tiens également à remercier Marc Girault, Louis Goubin et David Lubicz de me faire l'honneur de participer à mon jury, et par là même de juger ce travail.

Je remercie évidemment Louis Mahé d'avoir bien voulu m'encadrer dans le cadre de l'équipe de géométrie algébrique réelle de l'IRMAR. Sa mission fut particulièrement périlleuse, puisque j'ai rapidement choisi de m'éloigner de la géométrie algébrique. Sa patience, sa curiosité envers mes travaux et sa bienveillance m'ont été d'une grande aide.

J'adresse également tous mes remerciements à mon second directeur de thèse, Reynald Lercier, dont la contribution a été décisive pour me permettre de faire une thèse en parallèle d'un poste dans son équipe, au CÉLAR. Sa curiosité d'esprit m'a permis de « fuguer » loin de la géométrie algébrique, avant d'y revenir par un chemin détourné. Sa rigueur m'a guidé tout au long de mes travaux de recherche.

Je ne peux évidemment pas oublier ceux qui ont œuvré pour me permettre de faire une thèse durant mon cursus au CÉLAR, en particulier les sous-directeurs techniques (Marc Howyan et François Decourt), les chefs de la division SSI (Paul Pincemin et André Parriel), les chefs du département CRY (Didier Alquié, Olivier Mangeot et Sébastien Gryselyn). Leur soutien constant m'a permis de me lancer dans cette grande aventure, puis de la mener à son terme, et je les en remercie.

Bien sûr, tout au long de ce travail de recherche, les collègues ont une place prédominante. À l'IRMAR, à l'occasion de mes séjours réguliers, tous les vendredis, j'ai passé d'excellents moments dans le bureau, autour de problèmes mathématiques variés, partagés avec Valéry Mahé, Sylvain Brochard, Gweltaz Chatel, Clément Dunand, Lionel Chaussade, Colas Bardavid...

Au CÉLAR, le plaisir de partager mes problèmes avec les collègues s'est traduit par des situations très diverses, mais toujours enrichissantes. Naturellement, c'est plutôt vers le bureau occupé par David Lubicz et Stéphanie Alt que je dirigeais mes pas, suite à des thématiques proches, tant avec David qu'avec Stéphanie. Je crois que les murs résonnent encore des discussions engagées entre David et moi, mais c'était pour la bonne cause ! Les travaux menés conjointement avec David en sont la preuve. Le calme de Stéphanie et son approche pragmatique des problèmes m'ont bien souvent tiré de mauvais pas.

Au-delà de ces deux « squattés par défaut », j'ai pu apprécié la capacité d'écoute de Franck Landelle, Pierre Loidreau, Emmanuel Mayer, Didier Alquié et Vivien Dubois (sans oublier Anne Bourrée et Emmanuel Bresson, partis durant ma thèse), avec qui j'ai pu échanger quelques idées à de nombreuses reprises.

Je n'oublie pas Johann Barbier, seul membre de l'équipe à avoir suivi un cursus parallèle au mien : après une première rencontre lors du DEA d'Algorithmique il y a 6 ans, j'ai eu le plaisir de le retrouver lors de mon arrivée au CÉLAR. Seul autre thésard de l'équipe, c'est tout naturellement que nous avons affronté les mêmes difficultés diverses, et partagé les mêmes bonheurs lors de ces dernières années.

J'adresse également un grand merci à Valérie Rouat, que le hasard a bien voulu placer dans mon bureau (ou plus précisément l'inverse). Son soutien quotidien des plus petites tâches aux plus colossales, comme la dernière relecture de ce manuscrit, se fait toujours dans l'ombre, et avec un grand sourire. Je mesure ainsi chaque jour la chance d'avoir atterri en B242.

Je tiens à remercier collectivement les personnels administratifs et de soutien, tant de l'IRMAR que du CÉLAR, dont la contribution invisible n'en est pas moins indispensable aux travaux menés dans ces établissements.

Enfin, et à titre tout personnel, je remercie Émilie. Nos routes se sont croisées à l'été 2003, elles n'ont plus voulu se séparer depuis. La confiance qu'elle place en moi, son amour et son soutien sont mes meilleures motivations.

Table des matières

Introduction

La cryptographie, étymologiquement *écriture cachée*, a longtemps consisté en l'étude de la protection en confidentialité des communications entre deux acteurs. Il s'agit de « protéger » les informations échangées de telle sorte que la connaissance des transmissions ne permette pas à un tiers d'en extraire le sens.

L'absence de moyens de communication efficaces et peu coûteux a tout d'abord limité la cryptographie à des usages essentiellement diplomatiques pendant des siècles. Mais la révolution des télécommunications a provoqué le développement de cette science. Non seulement le nombre d'utilisateurs potentiels de la cryptographie a explosé, mais parallèlement de nouveaux besoins se sont créés. La cryptographie moderne ne consiste plus seulement à protéger des informations de la curiosité de tiers, mais aussi à fournir des garanties très diverses, dépendant du contexte d'utilisation. Les plus classiques sont :

- **l'authentification**, la garantie que la personne avec laquelle on échange des informations est bien celle avec laquelle on souhaite communiquer ;
- **la confidentialité**, la garantie que les informations échangées ne sont connues que de l'émetteur et du destinataire ;
- **l'intégrité**, la garantie que les messages échangés ne sont pas modifiés.

Le commerce électronique, par exemple, nécessite que le site marchand prouve son identité à l'usager, et que les communications lors du paiement soient confidentielles et intègres. En revanche, des services plus évolués, comme le vote électronique, font appel à de nombreuses garanties plus élaborées : ainsi, un électeur ne doit voter qu'une seule fois, il doit donc être identifié d'une certaine manière ; cet électeur doit avoir la certitude que son vote est pris en compte dans le scrutin ; simultanément, on doit assurer à l'électeur l'anonymat de son vote, c'est-à-dire qu'on ne doit pas pouvoir associer son vote et son identité.

Le problème considéré dans cette thèse est celui de la diffusion sécurisée vers de multiples utilisateurs, dont l'application principale réside dans la télévision à péage. Le contexte est le suivant : un émetteur souhaite transmettre de manière sécurisée et la plus efficace possible des informations identiques à de nombreux utilisateurs différents. L'objectif est que l'accès aux informations diffusées soit limité aux seuls utilisateurs prévus par l'émetteur. On décline cet objectif de sécurité en deux aspects bien différents.

- La protection contre les attaques externes : lorsque l'émetteur souhaite diffuser des informations, il choisit un ensemble de destinataires. Aucun utilisateur du système n'appartenant pas à cet ensemble (et par conséquent aucun tiers extérieur) ne doit pouvoir obtenir ces informations. On peut même demander que la protection soit valable contre des coalitions, c'est-à-dire contre plusieurs utilisateurs hors de cet ensemble, qui collaborent et disposent des secrets confiés à chacun d'eux. On parle alors de **schémas de diffusion**.
- Le traçage des attaques internes : même lorsqu'un schéma de diffusion robuste est utilisé, il est possible pour un destinataire de redistribuer les données diffusées. Plus simplement encore, un tel destinataire peut divulguer ses moyens de déchiffrement, et ainsi permettre à d'autres d'obtenir les informations de la même manière que lui. Dans ce cas, il est souhaitable de pouvoir identifier le destinataire ayant divulgué ses moyens de déchiffrement. S'il s'agit d'une coalition de destinataires, il faut pouvoir en identifier un membre. On parle alors de **schémas de traçage de traîtres**.

Diffusion par groupes

La problématique de la diffusion a été suggérée en 1993 dans [FN93]. Depuis, de nombreux travaux ont débouché sur des schémas de diffusion améliorant les performances ou traitant de cadres d'application différents.

Une première approche a conduit au schéma LKH (*logical key hierarchy*), proposé dans [WGL98, WHA99], et amélioré ultérieurement, dans [CGI+99, CMN99, PST01] notamment. L'idée de base est de faire en sorte que tous les destinataires prévus partagent une clé de déchiffrement commune : il suffit alors de chiffrer les données pour cette clé pour s'adresser exactement aux destinataires prévus. Chaque destinataire a connaissance de clés spécifiques, qui permettent le renouvellement de cette clé de déchiffrement commune lorsque l'ensemble des destinataires évolue. Cette catégorie de schémas est donc particulièrement adaptée lorsque l'ensemble des destinataires évolue peu. De plus, elle impose des exigences assez fortes sur les destinataires : ils doivent être en mesure de mémoriser durablement de nouvelles clés, et ils doivent être constamment à l'écoute d'éventuelles mises à jour de leurs clés.

Pour faire face à ces difficultés, mises en évidence dans [KRS99, GSW00], des schémas d'un nouveau type sont apparus : dans ces schémas, les seules clés durablement nécessaires aux destinataires sont des clés de déchiffrement qui leur ont été distribuées initialement. Ainsi, la non-réception de certains messages ne remet pas en cause la capacité des destinataires à obtenir les informations diffusées à long terme. Ces schémas, définis notamment dans [NNL01, HS02, GST04], déterminent des ensembles d'utilisateurs particuliers (le plus souvent à partir d'une structure en arbre binaire), et affectent à chaque ensemble une clé de déchiffrement particulière. Pour s'adresser à un ensemble de destinataires, l'émetteur utilise les clés associées à des ensembles recouvrant cet ensemble de destinataires. Ces schémas ne nécessitent pas de mise à jour des clés, mais la taille des chiffrés émis dépend de l'ensemble des destinataires. En pratique, les meilleurs

schémas garantissent des performances efficaces lorsque l'ensemble des destinataires est soit petit, soit proche de l'ensemble total des utilisateurs du système de diffusion.

L'utilisation de courbes elliptiques avec couplage a permis l'émergence de nouveaux résultats. Ainsi, avec [DF02], la taille de la clé publique a été considérablement réduite dans les schémas sans mémorisation de nouvelles clés. Un nouveau schéma à clé publique pour des utilisateurs sans mémoire a également été conçu dans [BGW05] à partir des couplages, et ce schéma est décrit comme ayant des clés privées et des messages chiffrés de tailles constantes (en fait, dans ce schéma, la clé publique est longue et nécessaire pour déchiffrer, et une description de l'ensemble des destinataires est requise dans les chiffrés).

Les couplages ouvrent donc de nouvelles perspectives : quelques schémas ont été proposés [GPSW06, BSW07, OSW07] pour gérer les utilisateurs en groupe, et permettre donc de s'adresser efficacement à des ensembles de destinataires de taille intermédiaire. Ces schémas sont cependant calculatoirement coûteux : nous proposons ici (et dans [LS08b]) un schéma de diffusion adapté à ce cadre. Des groupes d'utilisateurs sont définis de manière totalement libre lors de la distribution des clés de déchiffrement : pour un ensemble de destinataires s'exprimant simplement à partir des groupes d'utilisateurs, un mécanisme de chiffrement efficace peut être utilisé, quelle que soit la taille de cet ensemble de destinataires, et avec des calculs bien plus limités que dans les solutions précédentes.

Traçage de traîtres

Parallèlement, les bases du traçage de traîtres ont été posées en 1994 dans [CFN94]. Il ne s'agit pas ici de choisir un ensemble de destinataires particuliers : l'émetteur s'adresse à l'ensemble des utilisateurs du système. Par contre, lorsque des utilisateurs divulguent leurs clés de déchiffrement, et construisent un décodeur pirate à partir de ces clés, il doit être possible d'identifier l'un de ces utilisateurs par une analyse du décodeur pirate.

Différentes techniques ont été utilisées pour construire des schémas de traçage de traîtres. La première, utilisée notamment dans [CFN94, NP98, CFNP00], consiste à répartir de manière particulière des clés de déchiffrement indépendantes entre les utilisateurs : pour déchiffrer un message, un sous-ensemble de ces clés est nécessaire. L'identification des clés utilisées par un décodeur pirate permet de remonter jusqu'à un utilisateur ayant participé à sa construction.

Une seconde méthode, dédiée à la cryptographie asymétrique, consiste à générer des clés de déchiffrement corrélées, avec une structure commune. À partir d'un ensemble de clés de déchiffrement, une coalition peut alors construire des clés de déchiffrement nouvelles, c'est-à-dire différentes des clés utilisées par chacun de ses membres. Par contre, ces clés doivent doivent révéler les utilisateurs les ayant forgées. De nombreux schémas ont été proposés, notamment dans [KD98, BF99, MSK99, TSNZ03, MI04, BSW06, BW06], mais certains sont en fait vulnérables (en particulier ceux présentés dans [KD98, MSK99, TSNZ03]).

Une approche intermédiaire est proposée dans [KY02] : elle utilise un schéma asymétrique pour deux utilisateurs, et recombine ce schéma à de nombreuses reprises avec une bonne répartition des clés de déchiffrement. Cette dernière approche permet d'atteindre un ratio très performant entre les tailles d'un chiffré et du clair associé, allant même jusqu'à un ratio proche de 1 dans [CPP05].

Nous proposons ici (et dans [Sir07]) un nouveau schéma de traçage de traîtres. Sa conception suit l'approche définie dans [KY02], et ses performances sont donc similaires. Cependant, il est robuste face à des décodeurs pirates dotés de mémoire, et qui sont libres de ne pas répondre à certaines requêtes. Ce modèle de sécurité, qui renforce celui proposé dans [KY01a, KY01b], n'admettait aucun schéma avec de telles performances auparavant. Ce schéma présente également l'avantage de permettre aux destinataires de déchiffrer les messages progressivement : dans les schémas précédents à performance identique, on ne peut déchiffrer les données que par blocs de taille très importante.

Couplages définis par des courbes elliptiques

Une littérature extrêmement abondante s'est développée sur l'utilisation des couplages en cryptographie. Dans une première approche, les couplages ont été perçus comme un moyen d'attaquer des cryptosystèmes. Le schéma tripartite Diffie-Hellman proposé en 2000 dans [Jou00] est au contraire un schéma d'établissement de clé pour 3 intervenants, où l'existence d'un couplage apporte une fonctionnalité nouvelle et exploitable. Depuis, de très nombreuses constructions s'appuyant sur des couplages ont été proposées pour répondre à des problématiques cryptographiques diverses, la plus célèbre d'entre elles étant le chiffrement basé sur l'identité.

La construction de ces couplages suppose une bonne connaissance des points de torsion de courbes elliptiques définies sur un corps finis. C'est pourquoi nous nous intéressons tout d'abord ici (et dans [LS08a]) à cet aspect.

Par ailleurs, l'utilisation de couplages en cryptographie a nécessité la définition de nombreuses hypothèses calculatoires nouvelles, alors que ces hypothèses étaient en nombre assez limité en cryptographie asymétrique (typiquement la difficulté du problème RSA ou du logarithme discret dans des groupes classiques). Comment évaluer la validité de ces hypothèses ? Le modèle générique est un argument fort en faveur de ces hypothèses. Nous nous intéressons donc aussi ici (et dans [LS08c]) au modèle générique des groupes avec couplage, en faisant une présentation la plus rigoureuse possible, et en prouvant dans ce modèle les hypothèses de sécurité sur les groupes avec couplage les plus fréquentes. Ce modèle est notamment utilisé dans la preuve du schéma de diffusion évoqué précédemment.

Chapitre 1

Courbes elliptiques et ℓ-torsion en caractéristique quelconque

Les courbes elliptiques ont été introduites à la fin du XIX$^{\text{è}}$ siècle par Abel et Weierstrass. Leur nom provient de leur lien avec les intégrales elliptiques, mesurant la longueur d'un arc d'ellipse. Les courbes elliptiques sont particulièrement utilisées dans la cryptographie publique basée sur des problèmes proches du logarithme discret. En effet, alors qu'il est possible de calculer des logarithmes discrets avec des algorithmes sous-exponentiels dans les groupes multiplicatifs de corps finis, les seuls algorithmes connus pour calculer les logarithmes discrets sur des courbes elliptiques sont exponentiels. L'étude des propriétés des courbes elliptiques est donc nécessaire.

Le but de ce chapitre est d'étudier plus précisément le calcul des points de ℓ-torsion d'une courbe elliptique, et particulièrement dans le cas d'une caractéristique petite : des algorithmes efficaces existent pour le calcul de ces points en grande caractéristique, alors qu'en petite caractéristique, un tel calcul est bien plus coûteux, et les algorithmes existants considèrent des situations spécifiques.

Après avoir présenté le contexte des courbes elliptiques et des points de ℓ-torsion (partie 1.1), on présente quelques techniques existantes de calcul des points de ℓ-torsion en grande caractéristique (partie 1.2). On construit ensuite un nouvel algorithme de calcul des points de ℓ-torsion en grande caractéristique, basé sur les mêmes propriétés, mais dont la description est simplifiée (partie 1.3). Enfin, on étend ce dernier algorithme au cas d'une caractéristique quelconque, tout en préservant une bonne complexité, et nous illustrons la méthode avec quelques exemples (partie 1.4).

Le résultat principal de ce chapitre est donc :
« Pour tout corps de taille q et de caractéristique $p \geq 5$, pour tout ℓ premier impair différent de p, pour toute courbe définie sur ce corps, de type Elkies, l'algorithme donné calcule les polynômes minimaux des abscisses de deux points de ℓ-torsion indépendants en un nombre d'opérations élémentaires quasi-linéaire en $\ell \max(\ell, \log(q))^2$. »

1.1 Courbes elliptiques et ℓ-torsion

1.1.1 Introduction aux courbes elliptiques

La présentation suivante des courbes elliptiques et de divers outils pour ces courbes s'appuie sur [Sil86, GHM03, CFA+05].

1.1.1.1 Espace projectif et courbes

Soit $\mathbb{K}$ un corps. On définit la relation d'équivalence suivante sur $\mathbb{K}^3 \setminus \{(0,0,0)\}$: $(x_1, y_1, z_1) \sim (x_2, y_2, z_2)$ si et seulement si $\exists \lambda \in \mathbb{K}^* \,/\, x_1 = \lambda\, x_2,\ y_1 = \lambda\, y_2,\ z_1 = \lambda\, z_2$.

Définition 1.1 *L'espace projectif de dimension* 2 *sur* $\mathbb{K}$*, noté* $\mathbb{P}^2(\mathbb{K})$*, est l'ensemble quotient (ensemble des classes) de* $\mathbb{K}^3 \setminus \{(0,0,0)\}$ *par cette relation d'équivalence.*

Définition 1.2 *Une courbe projective plane de degré d sur* $\mathbb{K}$ *est l'ensemble des points* $(X : Y : Z)$ *de* $\mathbb{P}^2(\mathbb{K})$ *vérifiant une équation de type* $F(X, Y, Z) = 0$*, où* F *est un polynôme homogène de degré d à coefficients dans* $\mathbb{K}$*.*

Un point $(X_0 : Y_0 : Z_0)$ de la courbe projective plane associée au polynôme homogène F est dit singulier pour cette courbe si et seulement si

$$\frac{\partial F}{\partial X}(X_0 : Y_0 : Z_0) = \frac{\partial F}{\partial Y}(X_0 : Y_0 : Z_0) = \frac{\partial F}{\partial Z}(X_0 : Y_0 : Z_0) = 0.$$

À l'inverse, un point non-singulier est dit régulier.

Définition 1.3 *Une courbe elliptique est une courbe projective plane de degré* 3 *qui possède au moins un point dans* $\mathbb{P}^2(\mathbb{K})$ *et dont tous les points sont réguliers.*

1.1.1.2 Équations de Weierstrass, discriminant et j-invariant

Toute courbe elliptique est associée (par le biais de changements de variables) à une équation dite de Weierstrass :

$$F(X, Y, Z) = Y^2 Z + a_1\, XYZ + a_3\, YZ^2 - X^3 - a_2\, X^2 Z - a_4\, XZ^2 - a_6\, Z^3.$$

On en déduit qu'une courbe elliptique est constituée d'un point *à l'infini* $(0 : 1 : 0)$, noté P_∞, et d'une partie affine composée de points $(x : y : 1)$ vérifiant :

$$y^2 + a_1\, xy + a_3\, y = x^3 + a_2\, x^2 + a_4\, x + a_6.$$

Lorsque la caractéristique du corps $\mathbb{K}$ est différente de 2 et de 3, de nouveaux changements de variables permettent d'obtenir une équation dite de Weierstrass réduite :

$$F(X, Y, Z) = Y^2 Z - X^3 - a_4\, XZ^2 - a_6\, Z^3.$$

L'équation affine correspondante est alors :

$$y^2 = x^3 + a_4\, x + a_6.$$

On suppose désormais la caractéristique du corps $\mathbb{K}$ différente de 2 et de 3.

Définition 1.4 *On considère une courbe elliptique d'équation affine $y^2 = x^3 + a_4\,x + a_6$. Le discriminant de cette courbe elliptique, noté Δ, est défini par :*

$$\Delta = -16(4{a_4}^3 + 27{a_6}^2).$$

Propriété 1.1 *Le discriminant d'une courbe elliptique est non-nul.*

Preuve Par définition, une courbe elliptique est régulière. On utilise l'équation affine réduite de cette courbe : $f(x,y) = y^2 - x^3 - a_4\,x - a_6$. On suppose nul le discriminant de cette courbe, les coefficients de l'équation vérifient : ${a_6}^2 = -4{a_4}^3/27$.

On suppose a_4 non-nul. Soit $x_0 = -(3a_6)/(2a_4)$. Le point $(x_0, 0)$ appartient bien à la courbe :

$${x_0}^3 + a_4\,x_0 + a_6 = -\frac{27{a_6}^3}{8{a_4}^3} - \frac{3a_6}{2} + a_6 = 0.$$

Le point $(x_0, 0)$ est singulier pour cette courbe :

$$\frac{\partial f}{\partial x}(x_0, 0) = -3\,{x_0}^2 - a_4 = 0, \qquad \frac{\partial f}{\partial y}(x_0, 0) = 0.$$

Dans le cas où a_4 est nul, a_6 est également nul et le point $(0, 0)$ joue un rôle similaire : il appartient à la courbe et est singulier.

Par contradiction, le discriminant est non-nul. □

Définition 1.5 *On considère une courbe elliptique d'équation affine $y^2 = x^3 + a_4\,x + a_6$. Le j-invariant de cette courbe elliptique, noté j, est défini par :*

$$j = -12^3\,\frac{(4a_4)^3}{\Delta}.$$

Le j-invariant permet de caractériser certaines courbes :
- le j-invariant est nul si et seulement si a_4 est nul,
- le j-invariant est égal à $12^3 = 1728$ si et seulement si a_6 est nul.

En dehors de ces cas particuliers, un j-invariant caractérise des courbes elliptiques isomorphes en tant que variétés algébriques sur la clôture algébrique du corps $\mathbb{K}$. Plus précisément, en caractéristique différente de 2 et de 3, pour n'importe quelle valeur différente de 0 et de 1728, il existe deux courbes elliptiques ayant cette valeur pour j-invariant, à isomorphisme près sur le corps de base : une courbe elliptique différente avec le même j-invariant pourra se ramener à l'une des deux courbes elliptiques précédentes par un changement linéaire de variables. De plus, toujours lorsque la caractéristique est différente de 2 et de 3, dans une extension quadratique du corps de base, les deux courbes mentionnées précédemment sont isomorphes. Ainsi, dans une extension quadratique du corps de base, à isomorphisme près, un j-invariant correspond à une courbe elliptique unique.

1.1.1.3 Loi de groupe

Une loi peut être définie simplement à partir de la représentation géométrique d'une courbe elliptique. Dans un premier temps, l'opposé d'un point est défini par son symétrique par rapport à l'axe des abscisses. Ainsi, l'opposé du point (x, y) est le point $(x, -y)$.

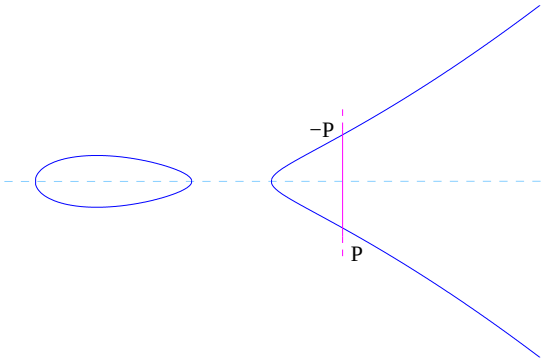

FIG. 1.1 – Opposé sur une courbe elliptique.

L'addition de deux points distincts s'appuie sur la droite passant géométriquement par ces deux points. Comme la courbe est une cubique non dégénérée, cette droite coupe la courbe en un troisième point, dont l'opposé est défini comme l'addition des deux points initiaux.

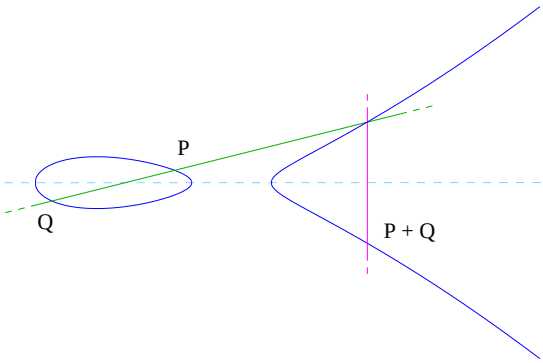

FIG. 1.2 – Addition sur une courbe elliptique.

Ainsi l'addition des deux points de coordonnées (x_1, y_1) et (x_2, y_2) est le point dont les coordonnées (x_3, y_3) sont données par :

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2, \\ y_3 = \lambda\,(x_1 - x_3) - y_1, \end{cases} \qquad \text{où} \quad \lambda = \frac{y_2 - y_1}{x_2 - x_1}\,.$$

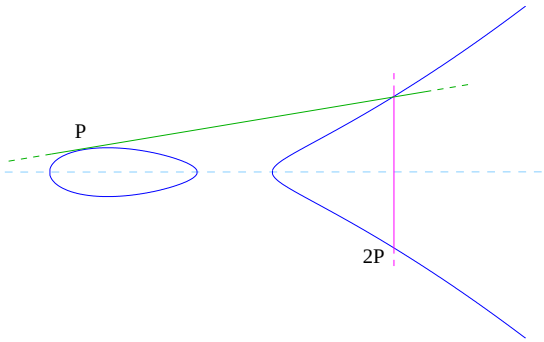

FIG. 1.3 – Duplication sur une courbe elliptique.

La duplication d'un point donné, c'est-à-dire son addition avec lui-même, s'appuie sur la tangente à la courbe en ce point. Cette tangente coupe la courbe en un troisième point, dont l'opposé est défini comme la duplication du point initial.

Ainsi la duplication d'un point de coordonnées (x, y) est le point dont les coordonnées (x', y') sont données par :

$$\begin{cases} x' = \lambda^2 - 2x, \\ y' = \lambda\,(x - x') - y, \end{cases} \qquad \text{où} \quad \lambda = \frac{3x^2 + a_4}{2y}\,.$$

Propriété 1.2 *Les opérations définies ainsi forment une loi de groupe, d'élément neutre P_∞, le point à l'infini de la courbe elliptique.*

1.1.1.4 Intérêt en cryptographie

Les courbes elliptiques sont utilisées en cryptographie dans des problèmes liés au calcul de logarithmes discrets : plusieurs hypothèses cryptographiques, découlant notamment des travaux de Diffie et Hellman présentés dans [DH76], nécessitent des groupes dans lesquels le calcul de logarithmes discrets doit être difficile.

Dans les groupes multiplicatifs des corps finis usuels ($\mathbb{F}_{p^n}^*$, avec p premier), le calcul de logarithmes discrets peut être réalisé par des algorithmes sous-exponentiels (voir par exemple [Gor93] pour le *number field sieve*, [Adl94] pour le *function field sieve*, [AD93a, AD93b] pour un algorithme sur tous les corps, et plus récemment [JL06b, JLSV06] pour une adaptation de ces cribles dans les cas intermédiaires). La taille des groupes doit donc être élevée pour assurer la difficulté du calcul.

Pour les courbes elliptiques, en revanche, les seuls algorithmes connus pour le calcul de logarithmes discrets sont exponentiels, si on écarte certaines courbes disposant de propriétés particulières. Ces algorithmes, dits *génériques* (voir chapitre 2), fonctionnent sur n'importe quel groupe. Et par conséquent, en l'absence d'algorithmes ayant une meilleure complexité, les courbes elliptiques fournissent une robustesse optimale au calcul du logarithme discret.

Concrètement, ces algorithmes génériques nécessitent un nombre d'opérations de l'ordre de $\sqrt{p}$, où p est le plus grand facteur premier de la taille du groupe considéré. Ainsi, le calcul de logarithmes discrets nécessite de l'ordre de 2^{80} opérations sur des courbes elliptiques bien choisies d'ordre premier p, de 160 bits. En comparaison (voir par exemple [LV00, LV01, Len04]), à même ordre de grandeur de temps de calculs, les corps finis usuels nécessitent des tailles supérieures à 1300 bits. L'intérêt des courbes elliptiques pour des applications cryptographiques apparaît ainsi clairement.

1.1.2 Problématique

1.1.2.1 Multiplication scalaire, polynôme de division

La formule d'addition pour une courbe elliptique sur $\mathbb{F}_q$, où $q = p^n$, est une fraction rationnelle des coordonnées des points impliqués dans l'addition. On en déduit que la multiplication par un scalaire ℓ, notée $[\ell]$, est également une fraction rationnelle à coefficients dans $\mathbb{F}_q$. On considère ce morphisme dans la clôture algébrique du corps de base, et on s'intéresse à son noyau, c'est-à-dire aux solutions de l'équation : $[\ell]P = P_\infty$.

Cette équation se traduit simplement par l'annulation du dénominateur des fractions rationnelles correspondant au morphisme $[\ell]$. Ce dénominateur peut s'écrire sous la forme d'un polynôme en l'abscisse du point, appelé polynôme de ℓ-division.

1.1.2.2 Objectif

On suppose que le scalaire ℓ est un nombre premier impair distinct de p. Dans ce cadre, l'ensemble des solutions de l'équation $[\ell]P = P_\infty$, appelé ensemble des points de ℓ-torsion et noté $E[\ell]$, est isomorphe à $(\mathbb{Z}/\ell\mathbb{Z})^2$ (voir par exemple [Sil86], page 89). Son cardinal est donc ℓ^2, et on en déduit que le polynôme de ℓ-division est de degré $(\ell^2-1)/2$. En effet, on écarte le point P_∞ qui n'apparaît pas dans les racines du polynôme, et une abscisse correspond à exactement deux points de ℓ-torsion.

L'objet de l'étude suivante est le calcul explicite des points de ℓ-torsion d'une courbe elliptique sur $\mathbb{F}_q$ donnée par une équation de Weierstrass. Plus exactement, on recherche les polynômes minimaux des abscisses de deux points de ℓ-torsion linéairement indépendants.

Ce problème est étroitement lié aux algorithmes de calcul d'isogénies. D'une part, la description des points de ℓ-torsion permet de calculer efficacement les isogénies d'ordre ℓ, via les formules de Vélu [Vél71]. D'autre part, les algorithmes rapides mentionnés ultérieurement s'appuient sur des calculs d'isogénies.

La résolution de ce problème est également une composante importante du calcul du nombre de points d'une courbe elliptique. En effet, par le biais du Frobenius, tous les points de la courbe elliptique sur la clôture algébrique $\overline{\mathbb{F}}_q$ vérifient une équation faisant intervenir le nombre de points de la courbe elliptique. En s'appuyant sur un point quelconque de la courbe, l'équation permet une recherche exhaustive du nombre

de points de la courbe dans un ensemble de $4\sqrt{q}$ éléments. En utilisant un point de ℓ-torsion, l'équation permet le calcul du nombre de points de la courbe modulo ℓ par une recherche exhaustive dans un ensemble de taille limitée à ℓ. En utilisant de nombreux ℓ différents et petits, dont le produit est supérieur à $4\sqrt{q}$, on identifie le nombre de points de la courbe de manière efficace.

1.2 Approches existantes

1.2.1 Notations

Pour comparer les approches présentées dans la suite, il est nécessaire d'évaluer le nombre d'opérations élémentaires réalisées dans chacune de ces approches. Plus que le nombre exact des opérations, on cherche en fait à majorer ce nombre, et à maîtriser l'évolution de ce nombre lorsque les paramètres augmentent. Ainsi, on utilise les notations O et $\tilde{O}$ suivantes dans le décompte des opérations élémentaires.

Définition 1.6 *Soient ϕ_1 et ϕ_2 deux fonctions de plusieurs variables réelles $(x_1, \ldots, x_k)$, à valeurs réelles. On dit que ϕ_1 est un « grand O » de ϕ_2 (en l'infini) s'il existe des constantes positives M, C_1, …, C_k telles que :*

$$\forall x_1 \geqslant C_1, \ldots, \forall x_k \geqslant C_k, \quad \phi_1(x_1, \ldots, x_k) \leqslant M\, \phi_2(x_1, \ldots, x_k).$$

Cette propriété est noté : $\phi_1(x_1, \ldots, x_k) = O(\phi_2(x_1, \ldots, x_k))$.

Cette notion permet de borner l'évolution d'une fonction à l'infini par celle d'une fonction de référence bien connue (exponentielle, puissance, logarithme…). En revanche, elle est trop précise pour certaines opérations, dites quasi-linéaires, dont le comportement à l'infini se rapproche de celui d'une fonction de type $x \log^n(x)$ pour un n donné.

Définition 1.7 *Soient ϕ_1 et ϕ_2 deux fonctions de plusieurs variables réelles $(x_1, \ldots, x_k)$, à valeurs réelles. On dit que ϕ_1 est un « grand O » à des facteurs logarithmiques près de ϕ_2 (en l'infini) s'il existe une constante n telles que :*

$$\phi_1(x_1, \ldots, x_k) = O\big(\phi_2(x_1, \ldots, x_k)\, \log^n\left(\phi_2(x_1, \ldots, x_k)\right)\big).$$

Cette propriété est noté : $\phi_1(x_1, \ldots, x_k) = \tilde{O}(\phi_2(x_1, \ldots, x_k))$.

Ces deux notions seront utilisées pour évaluer le nombre d'opérations requises par chacun des algorithmes étudiés, à des fins de comparaison. Ces notions n'étant significatives que du comportement à l'infini des fonctions, les comparaisons qui s'en déduisent ne sont valables que pour des valeurs suffisamment élevées des paramètres.

1.2.2 Approche naïve

Pour résoudre ce problème de calcul de points de ℓ-torsion, une approche directe existe, mais elle présente un coût calculatoire important.

Les polynômes minimaux recherchés correspondent naturellement à des facteurs du polynôme de ℓ-division. Une approche naïve consiste donc à calculer le polynôme de ℓ-division et à le factoriser.

La taille du polynôme de ℓ-division est de $O(\ell^2 \log(q))$ bits :
- Son calcul peut être réalisé en utilisant une approche « Square and Multiply » pour calculer formellement la multiplication par ℓ d'un point, ou des formules spécifiquement sur les dénominateurs qui s'en déduisent. Dans ce cas, un tel calcul nécessite $\tilde{O}(\ell^2 \log(q))$ opérations élémentaires.
- Sa factorisation utilise des techniques élaborées de calculs de plus grands diviseurs communs (voir notamment [Sho89]), dont le coût global peut être approximativement évalué à $\tilde{O}(\ell^{1,815\times 2} \log^2(q))$ opérations élémentaires.

Cet algorithme, séduisant par sa simplicité, nécessite donc environ $\tilde{O}(\ell^4 \log^2(q))$ opérations élémentaires.

1.2.3 Approche Elkies-Atkin

1.2.3.1 Algorithme

Cette technique s'appuie sur la structure très particulière des points de ℓ-torsion. En effet, l'ensemble des points de ℓ-torsion est stable non seulement par addition, mais également par l'action du Frobenius. Il en résulte que le polynôme de ℓ-division ne peut se factoriser que sous plusieurs formes particulières.

Ainsi, pour une courbe elliptique donnée, pour approximativement la moitié des valeurs de ℓ, le polynôme de ℓ-division admet deux facteurs différents de degré $(\ell-1)/2$. Ces facteurs se traduisent par l'existence de deux courbes elliptiques définies dans le corps de base, ℓ-isogènes à la courbe initiale.

Pour de telles valeurs de ℓ, on peut alors utiliser l'algorithme suivant :

1. On calcule le polynôme modulaire d'ordre ℓ, $\Phi_\ell(X, Y)$. Il s'agit d'un polynôme symétrique en deux variables, de degré $\ell + 1$ et dont les coefficients sont entiers et de longueur $O(\ell)$ bits. Ce polynôme traduit le fait que deux courbes elliptiques sont ℓ-isogènes : leurs j-invariants respectifs annulent le polynôme modulaire d'ordre ℓ.
2. À partir du j-invariant j_E de la courbe elliptique initiale, on calcule les deux j-invariants j_1 et j_2 racines dans le corps de base du polynôme $\Phi_\ell(X, j_E)$. Ces j-invariants correspondent aux deux courbes elliptiques sur le corps de base, ℓ-isogènes à la courbe initiale.
3. Pour chacun de ces deux j-invariants, on calcule l'équation de Weierstrass normalisée de la courbe elliptique correspondante, ainsi que la somme des abscisses des

points du noyau de l'isogénie, en utilisant les polynômes Φ_ℓ, $\partial\Phi_\ell/\partial X$, $\partial\Phi_\ell/\partial Y$, $\partial^2\Phi_\ell/\partial X^2$, $\partial^2\Phi_\ell/\partial X\partial Y$, $\partial^2\Phi_\ell/\partial Y^2$ (voir [Sch95]).

4. Pour chacune de ces courbes, on calcule l'isogénie, et on en déduit le polynôme de degré $(\ell-1)/2$ qui s'annule sur le noyau de l'isogénie : les deux polynômes obtenus sont les polynômes minimaux de deux points de ℓ-torsion linéairement indépendants.

1.2.3.2 Coût calculatoire en grande caractéristique

Étape 1.
Le polynôme modulaire $\Phi_\ell(X,Y)$ a $O(\ell^2)$ coefficients, de $O(\ell)$ bits chacun. Ce polynôme peut être calculé en un nombre d'opération quasi-linéaire en sa taille, d'après [Eng07], ce qui représente donc $\tilde{O}(\ell^3)$ opérations élémentaires.

La réduction modulo p de ce polynôme représente également $O(\ell^3)$ opérations élémentaires, et le polynôme résultant est un objet de $O(\ell^2\log(p))$ bits.

Étape 2.
En utilisant l'algorithme de Horner, l'évaluation du polynôme modulaire en j_E représente $\tilde{O}(l^2\log(q))$ opérations élémentaires. On obtient alors un polynôme de degré $l+1$, à coefficients dans $\mathbb{F}_q$.

Le calcul des racines utilise le calcul de plus grand dénominateur commun entre un facteur de ce polynôme et le polynôme X^q-X (voir [LN83]), ce qui représente $\tilde{O}(\ell\log^2(q))$ opérations. On obtient un polynôme de degré 2 en X dont les racines sont exactement les j-invariants recherchés.

Étape 3.
Les calculs de cette troisième étape utilisent des dérivées du polynôme modulaire sur $\mathbb{F}_q$. Les calculs des dérivées, et les évaluations qui suivent, peuvent être menés en $\tilde{O}(\ell^2\log(q))$ opérations.

Étape 4.
En utilisant l'algorithme de Charlap-Coley-Robbins [Mor95], le calcul de l'isogénie se ramène à la résolution d'une équation différentielle pour une fraction rationnelle dont le degré du numérateur est ℓ, et le degré du dénominateur est $\ell-1$. Pour identifier cette fraction rationnelle, un développement en série est utilisé : il suffit de calculer récursivement les 2ℓ premiers termes de cette série pour pouvoir reconstituer la fraction rationnelle.

Le calcul récursif direct des premiers termes représente $\tilde{O}(\ell^2\log(q))$, mais des techniques de calcul astucieuses basées sur des approximations successives permettent de ne réaliser que $\tilde{O}(\ell\log(q))$ opérations (d'après [BMSS07]). Le calcul de l'isogénie par cette résolution d'équation différentielle n'est en revanche possible qu'en grande caractéristique, c'est-à-dire lorsque ℓ est petit devant p.

Synthèse.
Quelle que soit la méthode choisie pour le calcul des isogénies, l'ensemble de l'algorithme représente globalement $\tilde{O}(\ell\max(\ell,\log(q))^2)$ opérations élémentaires lorsque la caractéristique du corps est grande.

1.2.4 Approches en petite caractéristique

En petite caractéristique, c'est-à-dire lorsque ℓ est plus grand que p, les étapes 1 à 3 sont identiques. L'étape 4 ne peut être réalisée comme précédemment, mais d'autres algorithmes existent :

- le premier algorithme est dû à Couveignes [Cou94], et il nécessite $\tilde{O}(\ell^3 \log(q))$ opérations élémentaires à p fixé ;
- un algorithme spécifique à la caractéristique 2 a été proposé par Lercier [Ler96], avec également $\tilde{O}(\ell^3 \log(q))$ opérations élémentaires, mais avec un gain significatif sur le facteur multiplicatif par rapport à l'algorithme précédent ;
- un troisième algorithme avec une meilleure complexité a été proposé par Couveignes [Cou96] : d'après des résultats présentés dans [Cou00], cet algorithme nécessite $\tilde{O}(\ell^2 \log(q))$ opérations élémentaires à p fixé.

Ces algorithmes ne sont efficaces que pour des valeurs très petites de p. En effet, l'algorithme de Lercier est spécifique à $p = 2$, et les algorithmes de Couveignes ont une complexité dépendant très fortement de p.

Un nouvel algorithme a été proposé par Joux et Lercier [JL06a] : il est valable pour toutes valeurs de p et ℓ, et sa complexité est raisonnable lorsque ℓ et p tendent simultanément vers l'infini. Leur approche repose sur l'utilisation des p-adiques, et elle nécessite $\tilde{O}((1 + \ell/p)\, \ell^2 \log(q))$ opérations élémentaires.

1.3 Nouvel algorithme en grande caractéristique

Dans l'optique d'un passage en petite caractéristique de la méthode présentée en 1.2.3, on présente dans un premier temps un algorithme en grande caractéristique, dont la transposition en caractéristique quelconque est facilement analysable.

Cet algorithme est une évolution des mécanismes décrits dans [BMSS07] : la stratégie générale est la même, mais la résolution de l'équation différentielle exploite quelques spécificités. Ainsi, l'algorithme obtenu n'a pas une meilleure complexité que l'algorithme décrit dans [BMSS07], mais il admet une description claire qui servira de support à notre analyse en caractéristique quelconque.

1.3.1 Isogénies

Définition 1.8 *Une isogénie est une application rationnelle non-nulle, d'une courbe elliptique E définie sur un corps $\mathbb{K}$ vers une courbe elliptique E' définie sur le même corps, qui envoie le point à l'infini de E vers celui de E'.*

D'après [Sil86], une isogénie est en particulier un morphisme de groupes. Le degré ℓ d'une isogénie est défini par le degré de l'extension $\mathbb{K}[E']/\mathbb{K}[E]$, où l'extension sur les corps de fonctions est induite par l'isogénie. Lorsque ℓ est premier avec la caractéristique du corps, l'isogénie est séparable et son noyau contient exactement ℓ éléments.

Exemple 1.1 *La multiplication d'un point de E par un scalaire ℓ (par rapport à la loi de groupe) est une isogénie, notée $[\ell] : E \to E$. Lorsque ℓ est premier avec la caractéristique du corps $\mathbb{K}$, le degré de cette isogénie est ℓ^2.*

Pour toute isogénie d'une courbe elliptique E vers E', il existe une isogénie unique de E' vers E, appelée isogénie duale, telle que la composition de ces isogénies soit $[\ell]$, c'est-à-dire la multiplication sur E par le degré de l'isogénie. Ainsi, le noyau d'une isogénie de degré ℓ est contenu dans l'ensemble des points de ℓ-torsion de E.

1.3.2 Forme particulière d'une isogénie

On considère désormais que la caractéristique du corps de base $\mathbb{K}$ est supérieure ou égale à 5. Une isogénie entre deux courbes elliptiques E et E', définies par des équations de Weiertrass, peut s'écrire sous la forme :

$$I(x, y) = (I_x(x), c\, y\, I'_x(x)).$$

Lorsque la constante c est égale à 1, l'isogénie est dite normalisée, et elle est en particulier séparable. Dans ce cas, d'après [Vél71], l'isogénie peut s'écrire sous la forme suivante :

$$I(x, y) = \left(\frac{N(x)}{D(x)}, y\left(\frac{N(x)}{D(x)}\right)'\right),$$

où N et D sont deux polynômes unitaires de degrés respectifs ℓ et $\ell-1$. De plus, lorsque ℓ est impair, D est le carré d'un polynôme g de degré $(\ell-1)/2$, polynôme minimal de points de ℓ-torsion de la courbe elliptique E.

À partir de cette forme particulière, on peut calculer une équation différentielle vérifiée par la fraction rationnelle $N(x)/D(x)$ à partir de la connaissance des courbes E et E'. Pour cela, il suffit d'écrire les équations de Weierstrass réduites des courbes E et E', et d'utiliser le fait que l'image par l'isogénie d'un point de E est un point de E'.

Ainsi, lorsque l'équation de Weierstrass réduite de E est $y^2 = x^3 + a_4\, x + a_6$, et lorsque celle de E' est $y^2 = x^3 + a'_4\, x + a'_6$, on obtient l'équation différentielle suivante :

$$(x^3 + a_4\, x + a_6)\left(\frac{N(x)}{D(x)}\right)'^2 = \left(\frac{N(x)}{D(x)}\right)^3 + a'_4\left(\frac{N(x)}{D(x)}\right) + a'_6. \tag{1.1}$$

1.3.3 Développement en Série

La résolution de cette équation différentielle en grande caractéristique passe classiquement par l'utilisation d'un développement en série. Ainsi, le développement en série de $N(x)/D(x) - x$ en la variable $1/x$ lorsque celle-ci tend vers 0 donne une formule de calcul par récurrence des coefficients de la série. Les premiers coefficients permettent donc de calculer une suite suffisamment grande de coefficients, un par un. On peut alors identifier D, puis N efficacement, en utilisant des techniques de reconstruction de fractions rationnelles comme l'algorithme de Berlekamp-Massey ou une variante optimisée.

Dans [BMSS07], la stratégie est similaire, mais le calcul des coefficients est réalisé plus efficacement, non pas en calculant les coefficients un à un, mais en utilisant des techniques d'approximations Newtoniennes : à chaque étape de calcul, on double le nombre de coefficients calculés.

Plus précisément, pour éviter des complications techniques, un changement de variable est effectué. Soit S définie par :

$$S(x) = \sqrt{\frac{D\,(1/x^2)}{N\,(1/x^2)}}\,, \text{ ou de manière équivalente } \frac{N(x)}{D(x)} = \frac{1}{S\left(\frac{1}{\sqrt{x}}\right)^2}.$$

L'équation (1.1) devient :

$$(a_6\,x^6 + a_4\,x^4 + 1)\left(S'(x)\right)^2 = 1 + a'_4\,S(x)^4 + a'_6\,S(x)^6.$$

En l'infini, $N(x)/D(x)$ a un développement limité de la forme : $x + O(1)$. Un développement de $S(x)$ en 0 s'en déduit : $S(x) = x + O(x^3)$. On obtient ainsi les conditions initiales sur S en 0 : $S(0) = 0$ et $S'(0) = 1$. Le calcul des premiers termes de la série S permet de reconstruire la fraction rationnelle $N(x)/D(x)$.

1.3.4 Résolution de l'équation différentielle

De manière plus générale par rapport à l'équation qu'on doit résoudre, on s'intéresse aux équations différentielles de la forme $S'^2 = G\cdot(H\circ S)$. En effet, l'équation à résoudre dans le paragraphe précédent correspond à $a'_6\,z^6 + a'_4\,z^4 + 1$ pour $H(z)$ et l'inverse de $a_6\,x^6 + a_4\,x^4 + 1$ pour $G(x)$. On recherche une solution sous la forme d'une fraction rationnelle, c'est-à-dire qu'on cherche à déterminer une solution modulo x^m, où m est préalablement donné. Le principe de la résolution est similaire à celui utilisé dans [BMSS07] : on suppose que la solution est connue modulo x^d, et on procède à une itération de Newton qui permet d'obtenir la solution modulo x^{2d}. En réitérant, on calcule globalement la solution à la précision voulue.

Dans cette section, on présente un algorithme de calcul, nécessitant $\tilde{O}(m\log(q))$ opérations élémentaires. Le théorème et la preuve justifient que l'algorithme calcule effectivement une solution de l'équation différentielle considérée.

Théorème 1.1 *Soit $(\alpha,\beta)\in\mathbb{K}^2$, soit G une série formelle sur $\mathbb{K}$, soit H un polynôme sur $\mathbb{K}$ tels que : $H(\alpha)=1$ et $\beta^2 = G(0)\neq 0$. On considère l'équation différentielle :*

$$S'(x)^2 = G(x)\,H(S(x))\,,\ S(0)=\alpha\,,\ S'(0)=\beta.$$

Soit car($\mathbb{K}$) la caractéristique de $\mathbb{K}$, soit $m\in\{1,\ldots,car(\mathbb{K})\}$. L'algorithme 1 calcule le développement en série (modulo x^m) de la solution de l'équation différentielle.

Algorithme 1 Résolution de l'équation « $S'^2 = G \cdot (H \circ S)\,,\ S(0) = \alpha\,,\ S'(0) = \beta$ »

Entrées : $m \in \mathbb{N}$, $(\alpha, \beta) \in \mathbb{K}^2$, $H \in \mathbb{K}[z]$, $G \in \mathbb{K}[[x]]$
Sorties : $S \in \mathbb{K}[x]$ solution de l'équation différentielle modulo x^m

$d \longleftarrow 2,\ \ U \longleftarrow 1/\beta,\ \ J \longleftarrow 1,\ \ V \longleftarrow 1$
$S \longleftarrow \alpha + \beta\, x + \left[\left(G'(0) + H'(\alpha)\,\beta^3\right)/(4\beta)\right] x^2$
tant que $(d < m - 1)$ **faire**
  $U \longleftarrow U \cdot \left(2 - S' \cdot U\right) \mod x^d$
  $V \longleftarrow \left(V + J \cdot (H \circ S) \cdot (2 - V \cdot J)\right) / 2 \mod x^d$
  $J \longleftarrow J \cdot \left(2 - V \cdot J\right) \mod x^d$
  $S \longleftarrow S + V \cdot \int \left(G \cdot (H \circ S) - S'^2\right) \left(U \cdot J\,/\,2\right) dx \mod x^{\min(2d+1,m)}$
  $d \longleftarrow 2d$
fin tant que
retourner S

Preuve Soit d un entier pair non nul. On suppose connue une série $S_d(x)$ solution de l'équation modulo x^{d+1}, c'est-à-dire :

$$S_d'^{\,2} = G \cdot (H \circ S_d) \mod x^d\,,\ S_d(0) = \alpha\,,\ S_d'(0) = \beta. \tag{1.2}$$

Soit $S_{2d} = S_d + A_{2d}$ solution de l'équation modulo x^{2d+1}, avec x^{d+1} divisant A_{2d}. On a alors : $(S_d' + A_{2d}')^2 = G \cdot \left(H \circ (S_d + A_{2d})\right) \mod x^{2d}$.

Cette équation se traduit par une équation différentielle linéaire en A_{2d} :

$$2\, S_d' \cdot A_{2d}' - G \cdot (H' \circ S_d) \cdot A_{2d} = G \cdot (H \circ S_d) - S_d'^{\,2} \mod x^{2d}.$$

Avec la condition initiale $A_{2d}(0) = 0$, la solution de cette équation est :

$$A_{2d} = \frac{1}{J_{2d}} \int \frac{\left(G \cdot (H \circ S_d) - S_d'^{\,2}\right) \cdot J_{2d}}{2\, S_d'}\, dx \mod x^{2d+1}\,, \tag{1.3}$$

$$\text{où} \quad J_{2d} = \exp\left(-\int \frac{G \cdot (H' \circ S_d)}{2\, S_d'}\, dx\right) \mod x^{2d+1}\,.$$

D'après l'équation (1.2), on sait que $\left(G \cdot (H \circ S_d) - S_d'^{\,2}\right)$ est divisible par x^d. Comme de plus S_d' a un coefficient constant non-nul, un facteur x^d va apparaître à l'intérieur de l'intégrale, dans le calcul de A_{2d}. Comme le contenu de cette intégrale doit être évalué modulo x^{2d}, il suffit donc de calculer J_{2d} modulo x^d.

L'inverse de J_{2d} apparaît à l'extérieur de cette intégrale : il sera multiplié par un facteur de x^{d+1} issu de l'intégrale, et il suffit donc d'évaluer cet inverse modulo x^d.

L'inverse de S_d' apparaît dans l'intégrale du calcul de A_{2d} et dans l'intégrale du calcul de J_{2d}. Dans le calcul de A_{2d}, il est multiplié par un facteur de x^d, et on en

calcule une primitive ensuite : un calcul modulo x^d est donc suffisant. Le calcul de J_{2d} s'effectuant seulement modulo x^d d'après le paragraphe précédent, le calcul de l'inverse de S'_d modulo x^d est globalement suffisant.

L'équation (1.2) donne une expression de l'inverse de S'_d modulo x^d :

$$\frac{1}{S'_d} = \frac{S'_d}{G \cdot (H \circ S_d)} \mod x^d.$$

En utilisant cette expression dans le calcul de l'intégrale utilisée dans J_{2d} modulo x^d, on obtient :

$$\begin{aligned} \int \frac{G \cdot (H' \circ S_d)}{2\, S'_d}\, dx &= \int \frac{S'_d \cdot (H' \circ S_d)}{2\,(H \circ S_d)}\, dx \mod x^d \\ &= \frac{\log(H \circ S_d)}{2} \mod x^d. \end{aligned}$$

On en déduit des formules simplifiées pour J_{2d} et $1/J_{2d}$ modulo x^d :

$$J_{2d} = \frac{1}{\sqrt{H \circ S_d}} \mod x^d\,, \quad \frac{1}{J_{2d}} = \sqrt{H \circ S_d} \mod x^d\,.$$

Ces formules permettent un calcul efficace de l'ensemble des éléments utiles pour l'évaluation de S_{2d}, à partir de S_d et des éléments nécessaires à son calcul, éléments supposés connus.

Ces éléments nécessaires au calcul de S_d sont :
- l'inverse de $S'_{d/2}$ modulo $x^{d/2}$, noté U_d,
- la valeur de $\sqrt{H \circ S_{d/2}}$ modulo $x^{d/2}$, notée V_d,
- l'inverse de V_d modulo $x^{d/2}$, noté J_d.

Le calcul de U_{2d}, inverse de S'_d modulo x^d, à partir de U_d et de S_d est réalisé par une itération de Newton classique pour l'inversion. En effet, comme $S_d = S_{d/2} \mod x^{d/2+1}$, on a $U_{2d} = U_d \mod x^{d/2}$, et on calcule la seconde moitié des coefficients de U_{2d} par la formule suivante :

$$U_{2d} = U_d \cdot \left(2 - S'_d \cdot U_d\right) \mod x^d.$$

Le calcul de V_{2d} consiste à calculer une solution en v modulo x^d de l'équation suivante : $v^2 - (H \circ S_d)(x) = 0$. À partir d'une solution V_d de cette équation modulo $x^{d/2}$, on peut calculer une solution modulo x^d en utilisant la formule suivante :

$$V_{2d} = \frac{1}{2}\left(V_d + \frac{H \circ S_d}{V_d}\right) \mod x^d.$$

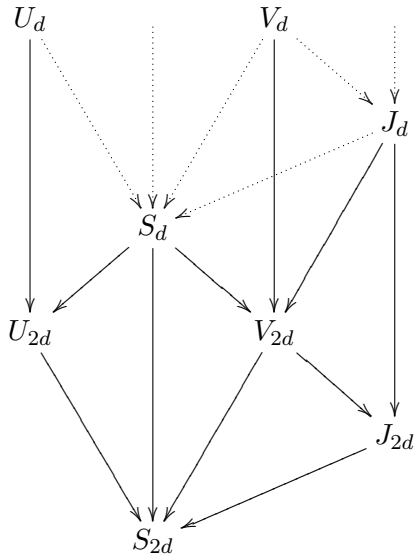

FIG. 1.4 – Succession des calculs de U_{2d}, V_{2d}, J_{2d} et S_{2d}

Il est nécessaire de calculer l'inverse de V_d modulo x^d. Or J_d et V_d sont par définition inverses modulo $x^{d/2}$. L'inverse W_{2d} de V_d modulo x^d est donc calculable par la formule de Newton de l'inversion déjà utilisée :

$$W_{2d} = J_d \cdot \big(2 - V_d \cdot J_d\big) \mod x^d.$$

En réinjectant cette valeur dans le calcul de V_{2d}, on obtient :

$$V_{2d} = \frac{1}{2}\Big(V_d + J_d \cdot (H \circ S_d) \cdot (2 - V_d \cdot J_d)\Big) \mod x^d.$$

Une nouvelle utilisation de la formule d'inversion permet le calcul de J_{2d} en fonction de J_d et de V_{2d} :

$$J_{2d} = J_d \cdot \big(2 - J_d \cdot V_{2d}\big) \mod x^d.$$

Les formules précédentes permettent donc de calculer (U_{2d}, V_{2d}, J_{2d}) à partir de (U_d, V_d, J_d, S_d). La valeur de S_{2d} s'obtient à partir de ces éléments, en utilisant l'équation (1.3) :

$$S_{2d} = S_d + \frac{V_{2d}}{2} \int U_{2d} \cdot J_{2d} \cdot \big(G \cdot (H \circ S_d) - S_d'^2\big)\, dx \mod x^{2d+1}.$$

Il reste à déterminer des valeurs initiales (pour $d = 2$) pour permettre des appels successifs à toutes ces formules, et donc le calcul global de la solution désirée. Soit γ défini par : $S_2(x) = \alpha + \beta\, x + \gamma\, x^2 \mod x^3$. La série S_2 vérifie l'équation différentielle,

et donc : $\beta^2 + 4\,\beta\,\gamma\,x = G(x)\,H(\alpha+\beta x) \mod x^2$. En dérivant, puis en posant $x=0$, on obtient la valeur de γ, et donc la valeur de S_2 :

$$S_2(x) = \alpha + \beta\,x + \left(\frac{G'(0)}{4\beta} + \frac{\beta^2\,H'(\alpha)}{4}\right) x^2 \mod x^3.$$

Par définition, $S_1(x) = \alpha + \beta\,x \mod x^2$. On en déduit :

$$U_2(x) = \frac{1}{\beta} \mod x\,, \quad V_2(x) = 1 \mod x \quad \text{et} \quad J_2(x) = 1 \mod x.$$

L'algorithme décrit réalise donc bien l'initialisation pour $d=2$. À chaque étape de la boucle, l'algorithme calcule les valeurs de $(U_{2d}, V_{2d}, J_{2d}, S_{2d})$ à partir des valeurs de (U_d, V_d, J_d, S_d) supposées données en entrée, puis double la valeur de d. Enfin, lorsque la précision d est suffisante, il renvoie la solution de l'équation différentielle. □

1.3.5 Algorithme complet

Pour calculer l'isogénie, dans un premier temps, on calcule $G(x)$, série inverse de $(a_6\,x^6 + a_4\,x^4 + 1)$ modulo $x^{4\ell-1}$. Pour cela, on utilise la formule récursive de calcul d'un inverse : $G_1(x) = 1$, $G_{2d}(x) = G_d(x)\left(2 - G_d(x)\cdot(a_6\,x^6 + a_4\,x^4 + 1)\right) \mod x^{2d}$. La série $G(x)$ s'obtient en réduisant modulo $x^{4\ell-1}$ la série $G_{d_0}(x)$, où d_0 est la puissance de 2 immédiatement supérieure ou égale à $4\ell-1$.

On définit alors $H(z) = a'_6\,z^6 + a'_4\,z^4 + 1$, puis on applique l'algorithme 1 de résolution de l'équation différentielle pour $m = 4\ell$, avec les paramètres additionnels suivants : $\alpha = 0$ et $\beta = 1$.

La solution S calculée précédemment ne contient que des termes de degré impair. On extrait ces coefficients de degré impair sous la forme d'une nouvelle série :

$$T(x) = \sum_{i=0}^{2\ell-1} t_i\,x^i\,, \qquad \text{où } \forall i \in \{0,\ldots,2\ell-1\}, t_i = s_{2i+1}.$$

On note $R(x)$ l'inverse du carré de $T(x)$, modulo $x^{2\ell}$. Cette série $R(x)$ est calculée en utilisant une formule récursive similaire à celle utilisée pour $G(x)$. En outre, elle vérifie l'équation suivante :

$$\frac{N(x)}{D(x)} = x\,R\left(\frac{1}{x}\right), \qquad \text{c'est-à-dire } R(x) = \frac{x^\ell\,N(1/x)}{x^{\ell-1}\,D(1/x)}.$$

Ainsi, $R(x)$ est le développement d'un quotient d'un polynôme de degré ℓ par un polynôme de degré $\ell-1$. Son développement modulo $x^{2\ell}$ permet une reconstruction par l'algorithme de Berlekamp-Massey [Ber68, Mas69, Dor87] ou une variante optimisée [BGY80, Pan00]. On déduit l'isogénie par l'identification des polynômes N et D, puis le polynôme minimal d'un point de ℓ-torsion en considérant la racine carrée de D.

Globalement, le calcul des polynômes minimaux des abscisses de deux points de ℓ-torsion indépendants, s'appuyant sur cette méthode de calcul des isogénies, présente les propriétés suivantes :

Théorème 1.2 *On considère l'algorithme de calcul des polynômes minimaux des abscisses de deux points de ℓ-torsion indépendants, décrit par les opérations présentées en paragraphe 1.2.3, par l'algorithme 1 du paragraphe 1.3.4 et par les opérations complémentaires de calcul d'isogénie données dans ce paragraphe.*

On considère le corps $\mathbb{F}_q$, où $q = p^n$ avec p premier différent de 2 et de 3. Cet algorithme est valide sur $\mathbb{F}_q$ pour tout nombre premier ℓ inférieur à $p/4$. Il nécessite alors $\tilde{O}(\ell \max(\ell, \log(q))^2)$ opérations élémentaires.

1.4 Algorithme en caractéristique quelconque

1.4.1 Principe

Pour obtenir une extension de l'algorithme de calcul d'isogénies d'Elkies-Atkin en caractéristique quelconque avec de bonnes performances, les techniques développées dans [JL06a] donnent l'idée générale : utiliser les p-adiques pour autoriser des divisions par la caractéristique du corps. La possibilité de réaliser ces divisions permet alors d'utiliser naturellement des algorithmes prévus pour ne fonctionner qu'en grande caractéristique. Il y a deux obstacles essentiels à cette approche :

1. les calculs réalisés dans les p-adiques doivent avoir un sens, et être interprétables dans le corps de base à la fin,
2. les calculs dans les p-adiques impliquent des pertes de précision lors des divisions par la caractéristique du corps. Il faut donc prévoir initialement une précision suffisante, qui se traduit par une augmentation de la taille des objets manipulés.

Dans un premier temps, on peut espérer réaliser ce passage dans les p-adiques seulement dans la dernière étape de l'algorithme, c'est-à-dire pour le calcul de l'isogénie. En fait, ce n'est pas possible : les algorithmes rapides de calcul d'isogénies en grande caractéristique utilisent le fait que l'isogénie issue des étapes précédentes est normalisée, et cette propriété n'est pas préservée lors du passage dans les p-adiques.

Il faut donc nécessairement passer dans les p-adiques dès le début de l'algorithme pour travailler sur des isogénies normalisées dans les p-adiques. C'est exactement la démarche utilisée dans [JL06a], avec une précision p-adique nécessaire linéaire en ℓ.

En s'appuyant sur les techniques calculatoires de [BMSS07], on montre en fait que la précision p-adique nécessaire au calcul de l'isogénie peut être ramenée à seulement $O(\log^2(\ell))$. La complexité globale de l'algorithme est alors similaire à celle des algorithmes en grande caractéristique, c'est-à-dire $\tilde{O}(\ell \max(\ell, \log(q))^2)$.

1.4.2 Utilisation des p-adiques

L'anneau $\mathbb{Z}_p$ des p-adiques est constitué de suites $(x_1, x_2, \ldots, x_k, \ldots)$ telles que :
– pour tout $k \in \mathbb{N}^*$, x_k est un élément de $\mathbb{Z}/p^k\mathbb{Z}$,
– pour tout $i \in \mathbb{N}^*$, pour tout $j > i$, la projection de x_j sur $\mathbb{Z}/p^i\mathbb{Z}$ est égale à x_i.

La structure d'anneau de $\mathbb{Z}_p$ est simplement déduite de celle des anneaux $\mathbb{Z}/p^k\mathbb{Z}$. Le corps $\mathbb{Q}_p$ des p-adiques est le corps des fractions de l'anneau $\mathbb{Z}_p$.

Pour obtenir une extension totalement non-ramifiée de degré n du corps $\mathbb{Q}_p$, on construit un relevé unitaire de degré n dans $\mathbb{Z}_p[X]$ du polynôme définissant l'extension $\mathbb{F}_{p^n}$ sur $\mathbb{F}_p$. Le polynôme résultant P est irréductible sur $\mathbb{Q}_p$ et permet de définir l'extension de degré n comme $\mathbb{Q}_p[X]/(P)$. On note $\mathbb{Q}_{p^n}$ cette extension.

D'un point de vue calculatoire, on ne peut pas travailler sur $\mathbb{Z}_p$, sur $\mathbb{Q}_p$ ou sur $\mathbb{Q}_{p^n}$, dont les éléments n'admettent pas de description finie. On doit limiter la précision p-adique. En pratique, pour travailler sur $\mathbb{Q}_p$ avec une précision p-adique maximale $w \in \mathbb{N}^*$ donnée, on utilise des représentations de la forme $x \cdot p^\lambda + O(p^\mu)$ où :
– λ et μ sont des entiers relatifs vérifiant $\lambda < \mu \leqslant \lambda + w$,
– x est un élément inversible de $\mathbb{Z}/p^{\mu-\lambda}\mathbb{Z}$.

On doit cependant rajouter des représentations de l'élément nul de $\mathbb{Q}_p$, qui s'écrivent sous la forme $O(p^\mu)$ où μ est un entier relatif.

Pour travailler sur $\mathbb{Z}_p$ avec une précision maximale $w \in \mathbb{N}^*$ donnée, on utilise les représentations précédentes de type $x \cdot p^\lambda + O(p^\mu)$, où λ doit être de plus un entier naturel, ainsi que les éléments précédents de la forme $O(p^\mu)$ où μ est un entier naturel. En ce qui concerne le corps $\mathbb{Q}_{p^n}$, on utilise des polynômes de degré inférieur à n dont les coefficients sont des représentations à précision w d'éléments de $\mathbb{Q}_p$.

Cette notion de précision caractérise la taille maximale nécessaire pour stocker une représentation : elle est donc utilisée en pratique pour des implémentations. En revanche, dès qu'on réalise des additions, la précision des représentations peut décroître de manière imprévisible. Pour éviter ce phénomène, on utilise une notion différente de précision, que l'on appelle précision absolue.

L'entier μ dans une représentation $x \cdot p^\lambda + O(p^\mu)$ ou $O(p^\mu)$ d'un élément de $\mathbb{Z}_p$ ou de $\mathbb{Q}_p$ est appelé précision absolue de la représentation. Dans $\mathbb{Q}_{p^n}$, la précision absolue d'une représentation d'un élément est définie par le minimum des précisions absolues de ses coefficients. Dans $\mathbb{Z}_p$, $\mathbb{Q}_p$ ou $\mathbb{Q}_{p^n}$, la précision absolue d'une somme de représentations est égale au minimum des précisions absolues des représentations additionnées.

Pour le cas de la multiplication, on ne considère que des représentations entières. Une représentation d'un élément de $\mathbb{Q}_p$ est dite entière si elle coïncide avec une représentation d'un élément de $\mathbb{Z}_p$. Une représentation d'un élément de $\mathbb{Q}_{p^n}$ est dite entière si ses coefficients sont des représentations entières. La précision absolue d'un produit de représentations entières de $\mathbb{Q}_p$ ou de $\mathbb{Q}_{p^n}$ est supérieure ou égale au minimum des précisions absolues des représentations multipliées.

1.4.3 Relèvement des courbes et des isogénies

On considère une courbe elliptique E définie sur $\mathbb{F}_{p^n}$, et un nombre premier ℓ différent de p. On suppose que le polynôme de ℓ-division de E admet deux facteurs différents de degré $(\ell-1)/2$ dans $\mathbb{F}_{p^n}$.

L'objectif est comme précédemment d'identifier ces deux facteurs de petit degré du polynôme de ℓ-division. Pour cela, on commence par relever arbitrairement la courbe E dans les p-adiques : $\tilde{a}_4 = a_4 \mod p$ et $\tilde{a}_6 = a_6 \mod p$. On travaille désormais sur la courbe elliptique $\tilde{E}$ sur $\mathbb{Q}_{p^n}$ d'équation : $y^2 = x^3 + \tilde{a}_4\, x + \tilde{a}_6$.

Le calcul du j-invariant $\tilde{j_E}$ de la courbe $\tilde{E}$, des solutions $\tilde{j_1}$ et $\tilde{j_2}$ de l'équation $\Phi_\ell(X, \tilde{j_E}) = 0$, ainsi que des équations de Weierstrass des courbes $\tilde{E}_1$ et $\tilde{E}_2$ correspondant à ces j-invariants se déroule exactement comme dans le cas de la grande caractéristique. Les courbes $\tilde{E}_1$ et $\tilde{E}_2$ sont ℓ-isogènes à la courbe $\tilde{E}$, et les isogénies peuvent être calculées comme en grande caractéristique.

La réduction E_1 de la courbe $\tilde{E}_1$ sur $\mathbb{F}_{p^n}$ est une courbe elliptique ℓ-isogène à la courbe E, et l'isogénie reliant ces deux courbes est simplement la réduction sur $\mathbb{F}_{p^n}$ de l'isogénie reliant $\tilde{E}$ à $\tilde{E}_1$. Il en est de même pour la réduction E_2 de la courbe $\tilde{E}_2$ sur $\mathbb{F}_{p^n}$. Il suffit donc de réduire les dénominateurs des isogénies sur le corps de base pour identifier les facteurs recherchés du polynôme de ℓ-division de E.

En pratique, à la fin des calculs, les représentations des coefficients des polynômes doivent être entières et de précision absolue non-nulle, pour être réductibles sur $\mathbb{F}_{p^n}$. On doit utiliser une précision p-adique suffisante lors du relèvement de la courbe E.

1.4.4 Calculs dans les entiers de $\mathbb{Q}_{p^n}$ et étude de la précision absolue

On s'intéresse désormais à la précision p-adique nécessaire lors du relèvement des courbes elliptiques : cette précision doit permettre qu'à la fin de la résolution de l'équation différentielle, la série produite puisse se réduire dans $\mathbb{F}_{p^n}$. Pour cela, le théorème suivant est particulièrement utile : si la précision p-adique initiale est suffisante, tous les calculs dans la résolution de l'équation différentielle portent sur des représentations entières et la perte de précision absolue engendrée est limitée.

Théorème 1.3 *Pour tout entier r non nul, soit $v_p(r)$ la plus grande puissance de p divisant r : $v_p(r) = \max\{k \in \mathbb{N} \,/\, p^k \text{ divise } r\}$. Pour tout entier i, on pose : $loop\_loss(p,\ell,i) = \max\{v_p(r) \,/\, 2^i + 1 \leqslant r \leqslant \min(2^{i+1}, 4\ell-1)\}$. On définit alors :*

$$loss(p,\ell) = \sum_{1 \leqslant i < \log_2(4\ell-1)} loop\_loss(p,\ell,i).$$

Lorsque la précision p-adique w_0 utilisée dans le relèvement est strictement supérieure à $loss(p,\ell)$, les représentations des coefficients des polynômes U, V, J et S sont entières à chacune des étapes de la résolution de l'équation différentielle (algorithme 1). De plus, les représentations des coefficients de la série S calculée par l'algorithme ont une précision absolue supérieure ou égale à $(w_0 - loss(p,\ell))$.

Preuve On prouve ce théorème par récurrence sur le nombre j de passages dans la boucle « *tant que* » de l'algorithme de résolution de l'équation différentielle.

Pour $0 \leqslant j < \log_2(4\ell-1)$, l'hypothèse de récurrence au rang j est : après j passages dans la boucle « *tant que* » de l'algorithme de résolution de l'équation différentielle, les représentations des coefficients des polynômes U, V, J et S sont entières et de précision absolue supérieure ou égale à $w_0 - \sum_{1 \leqslant i \leqslant j} \mathit{loop\_loss}(p, \ell, i)$.

INITIALISATION

Dans le calcul d'une isogénie, on utilise les paramètres suivants dans la résolution de l'équation différentielle : $\alpha = 0$, $\beta = 1$, $H(z) = \tilde{a}'_6\, z^6 + \tilde{a}'_4\, z^4 + 1$ et $G(x)$ série inverse de $\tilde{a}_6\, x^6 + \tilde{a}_4\, x^4 + 1$ modulo $x^{4\ell-1}$. Comme les représentations $\tilde{a}_4$, $\tilde{a}_6$, $\tilde{a}'_4$ et $\tilde{a}'_6$ sont entières de précision absolue w_0, les représentations des coefficients de G et H sont entières de précision absolue w_0 : c'est clair pour H, tandis que pour G, cela découle du calcul récursif d'une série inverse, où aucune division par p n'apparaît.

On en déduit que les représentations des coefficients des polynômes U, V, J et S initiaux sont entières de précision absolue égale à w_0 : l'initialisation est vérifiée.

HÉRÉDITÉ

Soit $j < \log_2(4\ell - 1)$, on suppose l'hypothèse vraie au rang $j-1$.

Lors du $j^{\text{ème}}$ passage dans la boucle de l'algorithme, les polynômes U, V et J sont mis à jour par additions, multiplications, dérivations et compositions des valeurs des polynômes U, V, J et S avant l'entrée dans la boucle. Toutes ces opérations préservent le caractère entier et la précision absolue des représentations des coefficients des polynômes. Les représentations des coefficients des polynômes U, V et J sont donc entières de précision absolue supérieure ou égale à $w_0 - \sum_{1 \leqslant i \leqslant j-1} \mathit{loop\_loss}(p, \ell, i)$.

En ce qui concerne S, toutes les opérations réalisées dans sa mise à jour préservent le caractère entier et la précision absolue des représentations de ses coefficients, à l'exception de l'intégration. Cette intégration engendre des divisions par p, et donc des représentations de coefficients potentiellement non-entières. L'intégration calcule les coefficients de la série de degré compris entre $2^j + 1$ et $\min(2^{j+1}, 4\ell - 1)$. La puissance de p la plus grande par laquelle on va réaliser une division est donc $\mathit{loop\_loss}(p, \ell, j)$. La précision absolue des représentations des coefficients de S est donc supérieure ou égale à $w_0 - \sum_{1 \leqslant i \leqslant j} \mathit{loop\_loss}(p, \ell, i)$, qui est strictement positif. Or les coefficients de la série S sont des relevés des coefficients de la série déduite de l'isogénie sur le corps de base. Comme la précision absolue des représentations des coefficients de S est strictement positive, ces représentations sont nécessairement réductibles, donc entières.

L'hypothèse est donc vraie au rang j : l'hérédité est vérifiée. □

Remarque 1.1 *Pour réduire la perte de précision, on peut utiliser dans l'algorithme le fait que la série S attendue est impaire. On ne calcule alors que les coefficients de degrés impairs dans l'intégrale, et on réduit la perte dans une boucle de l'algorithme à :*

$$\mathit{loop\_loss}'(p, \ell, i) = \max\left\{ v_p(2r+1) \,/\, 2^{i-1} \leqslant r \leqslant \min(2^i - 1, 2\ell - 1) \right\}.$$

La propriété suivante donne une borne asymptotique claire de la perte de précision maximale $loss(p, \ell)$ mentionnée dans le théorème 1.3.

Propriété 1.3 *Indépendamment de p, lorsque ℓ croît, $loss(p, \ell) = O\left(\log^2(\ell)\right)$. Plus précisément,*

$$loss(p, \ell) \leqslant \frac{(\log_2(4\ell - 1) + 1)^2}{\log_2(p)}.$$

Preuve Pour tout $i < \log_2(4\ell - 1)$, la valeur de $loop\_loss(p, \ell, i)$ est la puissance de p la plus élevée divisant un élément d'un ensemble d'entiers tous plus petits que 2^{i+1}, et donc strictement plus petits que $p^{\lfloor \log_p(2^{i+1}) \rfloor + 1}$. On en déduit que $loop\_loss(p, \ell, i) \leqslant \lfloor \log_p(2^{i+1}) \rfloor \leqslant \log_p(2^{i+1})$, et donc :

$$\begin{array}{rcl} loss(p, \ell) & \leqslant & \log_p(2) \left(\sum_{1 \leqslant i < \log_2(4\ell-1)} (i + 1) \right) \\ & \leqslant & \log_p(2) \ (\log_2(4\ell - 1) + 1) \ (\log_2(4\ell - 1) + 2) \ / \, 2 \\ & \leqslant & \log_p(2) \ (\log_2(4\ell - 1) + 1)^2 \\ & \leqslant & (\log_2(4\ell - 1) + 1)^2 \, / \log_2(p). \end{array}$$

Ainsi lorsque ℓ croît, indépendamment de la valeur de p, $loss(p, \ell) = O\left(\log^2(\ell)\right)$. □

Corollaire 1.1 *Une précision p-adique de l'ordre de $\log^2(\ell)/\log(p)$ est suffisante pour calculer le polynôme minimal d'un point de ℓ-torsion en caractéristique quelconque, en suivant un cheminement similaire à celui décrit en section 1.3.5, et en utilisant l'algorithme de résolution d'équations différentielles mentionné en section 1.3.4. Ainsi, le calcul d'un polynôme minimal d'un point de ℓ-torsion nécessite $\tilde{O}(\ell \max(\ell, \log(q))^2)$ opérations.*

Preuve La première partie du corollaire découle directement du théorème 1.3 et de la propriété 1.3.

L'évaluation de la complexité du calcul du polynôme minimal d'un point de ℓ-torsion s'obtient aisément : les calculs réalisés dans les p-adiques sont basées sur des objets mathématiques de taille multipliée par un facteur de l'ordre de $\log^2(\ell)$ par rapport au calcul en grande caractéristique. Comme toutes ces calculs sont polynomiaux, l'augmentation de la taille des objets mathématiques induit un facteur polynomial en $\log(\ell)$ dans la complexité globale, qui est donc bien pris en compte dans les $\tilde{O}(l \max(l, \log(q))^2)$ opérations annoncées.

En ce qui concerne les calculs réalisés après retour dans le corps de base, c'est-à-dire juste après le calcul de la solution de l'équation différentielle, ils sont exactement similaires à leurs homologues en grande caractéristique. □

La figure 1.5 montrent concrètement l'évolution de la précision nécessaire lorsque p et ℓ varient. La précision théorique mentionnée correspond au calcul issu de la perte

Précision nécessaire pour $p = 5$:

Valeur de ℓ	7	11	13	17	19 - 31	37	41 - 61	67	71 - 89	97
Pratique	5	6	6	7	8	11	11	13	13	14
Théorique	5	6	7	8	9	11	12	14	15	16

Précision nécessaire pour $p = 7$:

Valeur de ℓ	11	13	17	19 - 23	29 - 31	37 - 61	67 - 73	79 - 83	89 - 97
Pratique	4	5	6	6	6	8	10	10	11
Théorique	5	6	6	7	8	10	11	12	13

Précision nécessaire pour $p = 11$:

Valeur de ℓ	13	17 - 29	31	37 - 59	61	67 - 89	97
Pratique	3	4	5	6	6	7	8
Théorique	4	5	6	7	8	9	10

FIG. 1.5 – Précisions pratiques et théoriques pour $p \in \{5, 7, 11\}$ et $\ell \leqslant 97$

de précision $loss(p, \ell)$ mentionnée précédemment. La précision pratique mentionnée est celle qui semble nécessaire lors des calculs (vérifiée sur quelques exemples).

La précision nécessaire en pratique croît plus lentement que la borne théorique calculée, mais pour de petites valeurs de ℓ, l'écart entre ces deux précisions reste faible. La figure 1.6 suivante montre l'évolution de ces précisions en fonctions du logarithme de ℓ pour les mêmes valeurs de p : des sauts sont visibles, mais la croissance de la précision nécessaire reste globalement limitée.

	Valeur de ℓ	11	17	37	67	131	257
	Valeur de $\lfloor \log_2(\ell) \rfloor$	3	4	5	6	7	8
$p = 5$	Pratique	6	7	11	13	16	21
	Théorique	6	8	11	14	17	22
$p = 7$	Pratique	4	6	8	10	13	15
	Théorique	5	6	10	11	14	16
$p = 11$	Pratique	×	4	6	7	9	12
	Théorique	×	5	7	9	11	12

FIG. 1.6 – Précision nécessaire pour $\log_2(\ell) \lesssim 8$

1.4.5 Exemple détaillé sur une petite extension

Dans cette section, on vérifie la technique de calcul donnée précédemment via un exemple sur $\mathbb{F}_5$. On considère la courbe d'équation $y^2 = x^3 + x + 4$ sur $\mathbb{F}_5$. On s'intéresse au cas $\ell = 11$.

Pour calculer la précision 5-adique nécessaire, on évalue les pertes à chacune des étapes de la résolution de l'équation différentielle :

$$\begin{array}{lll} \mathit{loop\_loss}(5,11,1)=0, & \mathit{loop\_loss}(5,11,2)=1, & \mathit{loop\_loss}(5,11,3)=1, \\ \mathit{loop\_loss}(5,11,4)=2, & \mathit{loop\_loss}(5,11,5)=1. & \end{array}$$

On en déduit : $\mathit{loss}(5,11)=5$. La précision 5-adique nécessaire est donc 6.

On relève arbitrairement la courbe sur les 5-adiques, avec précision 6. On obtient l'équation suivante : $y^2 = x^3 + x + 4 + O(5^6)$.

Le calcul du polynôme modulaire et de ses dérivées partielles permet d'identifier l'équation de Weierstrass réduite d'une courbe ℓ-isogène : $y^2 = x^3 - (7329 + O(5^6))x - (3934 + O(5^6))$. On obtient donc les paramètres suivants :
$\tilde{a}_4 = 1 + O(5^6)\,,\ \tilde{a}_6 = 4 + O(5^6)\,,\ \tilde{a}'_4 = -7329 + O(5^6)\,,\ \tilde{a}'_6 = -3934 + O(5^6)$.

On peut alors calculer la série $\tilde{G}(x)$ modulo $x^{4\ell-1}$:

$$\begin{aligned} \tilde{G}(x) \;=\;& \big(4374+O(5^6)\big)\,x^{42} + \big(O(5^6)\big)\,x^{41} + \big(4298+O(5^6)\big)\,x^{40} + \big(O(5^6)\big)\,x^{39} \\ & - \big(2331+O(5^6)\big)\,x^{38} + \big(O(5^6)\big)\,x^{37} - \big(4417+O(5^6)\big)\,x^{36} + \big(O(5^6)\big)\,x^{35} \\ & + \big(3936+O(5^6)\big)\,x^{34} + \big(O(5^6)\big)\,x^{33} + \big(3505+O(5^6)\big)\,x^{32} + \big(O(5^6)\big)\,x^{31} \\ & + \big(228+O(5^6)\big)\,x^{30} + \big(O(5^6)\big)\,x^{29} - \big(1041+O(5^6)\big)\,x^{28} + \big(O(5^6)\big)\,x^{27} \\ & - \big(616+O(5^6)\big)\,x^{26} + \big(O(5^6)\big)\,x^{25} + \big(97+O(5^6)\big)\,x^{24} + \big(O(5^6)\big)\,x^{23} \\ & + \big(236+O(5^6)\big)\,x^{22} + \big(O(5^6)\big)\,x^{21} + \big(95+O(5^6)\big)\,x^{20} + \big(O(5^6)\big)\,x^{19} \\ & - \big(48+O(5^6)\big)\,x^{18} + \big(O(5^6)\big)\,x^{17} - \big(47+O(5^6)\big)\,x^{16} + \big(O(5^6)\big)\,x^{15} \\ & - \big(12+O(5^6)\big)\,x^{14} + \big(O(5^6)\big)\,x^{13} + \big(15+O(5^6)\big)\,x^{12} + \big(O(5^6)\big)\,x^{11} \\ & + \big(8+O(5^6)\big)\,x^{10} + \big(O(5^6)\big)\,x^{9} + \big(1+O(5^6)\big)\,x^{8} + \big(O(5^6)\big)\,x^{7} \\ & - \big(4+O(5^6)\big)\,x^{6} + \big(O(5^6)\big)\,x^{5} - \big(1+O(5^6)\big)\,x^{4} + \big(O(5^6)\big)\,x^{3} \\ & + \big(O(5^6)\big)\,x^{2} + \big(O(5^6)\big)\,x + \big(1+O(5^6)\big) \mod x^{43}\,. \end{aligned}$$

On résout ensuite l'équation différentielle basée sur $\tilde{H}(z) = \tilde{a}'_6\,z^6 + \tilde{a}'_4\,z^4 + 1$ et sur $\tilde{G}(x)$, en utilisant l'algorithme. On obtient la solution $\tilde{S}(x)$ modulo $x^{4\ell}$:

$$\begin{aligned} \tilde{S}(x) \;=\;& -(2+O(5))\,x^{43} + \big(O(5^3)\big)\,x^{42} + (2+O(5))\,x^{41} + \big(O(5^3)\big)\,x^{40} \\ & - (1+O(5))\,x^{39} + \big(O(5^3)\big)\,x^{38} + \big(8+O(5^2)\big)\,x^{37} + \big(O(5^3)\big)\,x^{36} \\ & - (1+O(5))\,x^{35} + \big(O(5^3)\big)\,x^{34} + \big(O(5^2)\big)\,x^{33} + \big(O(5^4)\big)\,x^{32} \\ & - \big(12+O(5^2)\big)\,x^{31} + \big(O(5^4)\big)\,x^{30} - \big(10+O(5^2)\big)\,x^{29} + \big(O(5^4)\big)\,x^{28} \\ & - \big(7+O(5^2)\big)\,x^{27} + \big(O(5^4)\big)\,x^{26} - \big(1+O(5^2)\big)\,x^{25} + \big(O(5^4)\big)\,x^{24} \\ & + \big(192+O(5^4)\big)\,x^{23} + \big(O(5^4)\big)\,x^{22} + \big(125+O(5^4)\big)\,x^{21} + \big(O(5^4)\big)\,x^{20} \\ & + \big(293+O(5^4)\big)\,x^{19} + \big(O(5^5)\big)\,x^{18} + \big(4+O(5^4)\big)\,x^{17} + \big(O(5^5)\big)\,x^{16} \\ & - \big(161+O(5^4)\big)\,x^{15} + \big(O(5^5)\big)\,x^{14} - \big(611+O(5^5)\big)\,x^{13} + \big(O(5^5)\big)\,x^{12} \\ & + \big(211+O(5^5)\big)\,x^{11} + \big(O(5^5)\big)\,x^{10} - \big(1494+O(5^5)\big)\,x^{9} + \big(O(5^6)\big)\,x^{8} \\ & + \big(1058+O(5^5)\big)\,x^{7} + \big(O(5^6)\big)\,x^{6} - \big(733+O(5^5)\big)\,x^{5} + \big(O(5^6)\big)\,x^{4} \\ & + \big(O(5^6)\big)\,x^{3} + \big(O(5^6)\big)\,x^{2} + \big(1+O(5^6)\big)\,x + \big(O(5^6)\big) \\ & \mod x^{44}\,. \end{aligned}$$

On réduit modulo 5 cette série :

$$\begin{aligned} S(x) \;=\;& 3\,x^{43} + 2\,x^{41} + 4\,x^{39} + 3\,x^{37} + 4\,x^{35} + 3\,x^{31} + 3\,x^{27} + 4\,x^{25} + 2\,x^{23} \\ & + 3\,x^{19} + 4\,x^{17} + 4\,x^{15} + 4\,x^{13} + x^{11} + x^{9} + 3\,x^{7} + 2\,x^{5} + x \mod x^{44}\,. \end{aligned}$$

On extrait les coefficients impairs, qui donnent $T(x)$ modulo $x^{2\ell}$:

$$\begin{aligned} T(x) \;=\; & 3\,x^{21} + 2\,x^{20} + 4\,x^{19} + 3\,x^{18} + 4\,x^{17} + 3\,x^{15} + 3\,x^{13} + 4\,x^{12} + 2\,x^{11} \\ & + 3\,x^{9} + 4\,x^{8} + 4\,x^{7} + 4\,x^{6} + x^{5} + x^{4} + 3\,x^{3} + 2\,x^{2} + 1 \mod x^{22}. \end{aligned}$$

On obtient $R(x)$ comme inverse de $T(x)^2$ modulo $x^{2\ell}$:

$$\begin{aligned} R(x) \;=\; & 2\,x^{20} + 2\,x^{19} + 3\,x^{18} + x^{16} + 2\,x^{15} + 3\,x^{14} + x^{13} + 3\,x^{12} + 2\,x^{11} \\ & + 2\,x^{10} + 2\,x^{8} + 3\,x^{7} + 4\,x^{6} + 4\,x^{5} + 4\,x^{3} + x^{2} + 1 \mod x^{22}. \end{aligned}$$

La reconstruction de la fraction rationnelle dont R est le développement modulo $x^{2\ell}$ donne les résultats suivants :

$$R(x) \;=\; \frac{3\,x^{11} + x^{9} + x^{8} + x^{7} + x^{6} + 3\,x^{5} + 2\,x^{4} + 3\,x^{3} + 2\,x^{2} + 2\,x + 1}{x^{10} + x^{9} + x^{8} + x^{7} + 3\,x^{6} + 3\,x^{5} + 3\,x^{4} + 2\,x^{3} + x^{2} + 2\,x + 1} \mod x^{22}.$$

On inverse l'ordre des coefficients du dénominateur pour obtenir $D(x)$:

$$D(x) \;=\; x^{10} + 2\,x^{9} + x^{8} + 2\,x^{7} + 3\,x^{6} + 3\,x^{5} + 3\,x^{4} + x^{3} + x^{2} + x + 1.$$

Le polynôme minimal d'un point de ℓ-torsion est la racine carrée de $D(x)$:

$$\sqrt{D(x)} \;=\; x^{5} + x^{4} + x^{2} + 3\,x + 1.$$

1.4.6 Exemple dans le cas général

Dans cette section, on vérifie la technique de calcul donnée précédemment via un exemple sur $\mathbb{F}_{7^5}$ (l'extension est définie par un élément ξ vérifiant $\xi^5 + \xi + 4 = 0$). On s'intéresse au cas $\ell = 47$, et on considère la courbe d'équation :

$$y^2 = x^3 + (5\,\xi^3 + 2\,\xi^2 + 4\,\xi + 6)\,x + (6\,\xi^4 + 4\,\xi^3 + 5\,\xi^2 + 3\,\xi + 4).$$

Pour calculer la précision nécessaire, on évalue les pertes à chacune des étapes de la résolution de l'équation différentielle :

$$\begin{array}{lll} \mathit{loop\_loss}(7,47,1) = 0, & \mathit{loop\_loss}(7,47,2) = 1, & \mathit{loop\_loss}(7,47,3) = 1, \\ \mathit{loop\_loss}(7,47,4) = 1, & \mathit{loop\_loss}(7,47,5) = 2, & \mathit{loop\_loss}(7,47,6) = 2, \\ \mathit{loop\_loss}(7,47,7) = 2. & & \end{array}$$

On en déduit : $\mathit{loss}(7,47) = 9$. La précision théorique nécessaire est donc 10. On relève arbitrairement la courbe avec cette précision, et on obtient les coefficients suivants :

$$\begin{aligned} \tilde{a}_4 &= 5\,\xi^3 + 2\,\xi^2 + 4\,\xi + 6 + O(7^{10}), \\ \tilde{a}_6 &= 6\,\xi^4 + 4\,\xi^3 + 5\,\xi^2 + 3\,\xi + 4 + O(7^{10}). \end{aligned}$$

Le calcul du polynôme modulaire et de ses dérivées partielles permet d'identifier l'équation de Weierstrass réduite d'une courbe ℓ-isogène : $y^2 = x^3 + \tilde{a}'_4\,x + \tilde{a}'_6$, avec

$$\begin{aligned} \tilde{a}'_4 &= -104574295\,\xi^4 - 111798340\,\xi^3 - 21387164\,\xi^2 - 24214869\,\xi + 36208471 + O(7^{10}), \\ \tilde{a}'_6 &= 88497100\,\xi^4 + 47971900\,\xi^3 + 32578586\,\xi^2 + 122102312\,\xi - 83236646 + O(7^{10}). \end{aligned}$$

La résolution de l'équation différentielle modulo $x^{4\ell}$ donne la solution suivante :

$$
\begin{aligned}
&\big(-64\,\xi^4+2\,\xi^3-156\,\xi^2+29\,\xi-167+O(7^3)\big)\,x^{187}+\big(O(7^4)\big)\,x^{186}\\
&+\big(-40\,\xi^4-16\,\xi^3-19\,\xi^2+31\,\xi+119+O(7^3)\big)\,x^{185}+\big(O(7^4)\big)\,x^{184}\\
&+\big(4\,\xi^4-135\,\xi^3+130\,\xi^2-71\,\xi-60+O(7^3)\big)\,x^{183}+\big(O(7^4)\big)\,x^{182}\\
&+\big(54\,\xi^4+106\,\xi^3+142\,\xi^2+34\,\xi+31+O(7^3)\big)\,x^{181}+\big(O(7^4)\big)\,x^{180}\\
&+\big(90\,\xi^4+68\,\xi^3-86\,\xi^2+85\,\xi-154+O(7^3)\big)\,x^{179}+\big(O(7^4)\big)\,x^{178}\\
&+\big(-90\,\xi^4-164\,\xi^3+148\,\xi^2+161\,\xi+109+O(7^3)\big)\,x^{177}+\big(O(7^4)\big)\,x^{176}\\
&+\ \ldots\\
&+\big(-14495715\,\xi^4-4997332\,\xi^3-13875420\,\xi^2+15139718\,\xi-6889283+O(7^9)\big)\,x^{11}\\
&+\big(-10316162\,\xi^4-18413483\,\xi^3+10203681\,\xi^2+12042174\,\xi-16856936+O(7^9)\big)\,x^{9}\\
&+\big(6321221\,\xi^4+3426564\,\xi^3-17849762\,\xi^2-11455210\,\xi-5945475+O(7^9)\big)\,x^{7}\\
&+\big(130780195\,\xi^4+130057790\,\xi^3+110851383\,\xi^2-87164062\,\xi-137616778+O(7^{10})\big)\,x^{5}\\
&+\big(1+O(7^{10})\big)\,x \mod x^{188}\,.
\end{aligned}
$$

On remarque en particulier que la perte de précision est plus faible que celle prévue théoriquement : les termes de plus haut degré ont une précision absolue de 3, alors qu'une précision absolue de 1 suffirait à réduire sur le corps de base. Pour cet exemple, la précision 7-adique réellement nécessaire au calcul est donc 8.

Le polynôme minimal d'un point de ℓ-torsion calculé par l'algorithme s'en déduit :

$$
\begin{aligned}
&x^{23}+(3\,\xi^4+5\,\xi^3+5\,\xi^2+6\,\xi+3)\,x^{22}\\
&+(5\,\xi^4+2\,\xi^2+3\,\xi+4)\,x^{21}+(3\,\xi^4+5\,\xi^3+2\,\xi^2+2\,\xi)\,x^{20}\\
&+(5\,\xi^4+5\,\xi^3+3\,\xi^2+6\,\xi+2)\,x^{19}+(6\,\xi^4+\xi^3+6\,\xi+4)\,x^{18}\\
&+(3\,\xi^4+5\,\xi^3+4\,\xi^2+4\,\xi+6)\,x^{17}+(6\,\xi^3+4\,\xi+2)\,x^{16}\\
&+(5\,\xi^4+6\,\xi^3+4\,\xi^2+\xi)\,x^{15}+(4\,\xi^4+5\,\xi^2+2)\,x^{14}\\
&+(3\,\xi^4+5\,\xi^2+5\,\xi+6)\,x^{13}+(\xi^4+3\,\xi^3+2\,\xi^2+6\,\xi+4)\,x^{12}\\
&+(\xi^4+3\,\xi^3+\xi^2+2\,\xi+4)\,x^{11}+(2\,\xi^4+\xi^3+6\,\xi^2+4\,\xi+5)\,x^{10}\\
&+(\xi^4+3\,\xi^3+\xi+6)\,x^{9}+(3\,\xi^4+\xi^3+4\,\xi+1)\,x^{8}\\
&+(6\,\xi^4+\xi^3+4\,\xi^2+\xi+5)\,x^{7}+(2\,\xi^3+5\,\xi^2+6\,\xi+6)\,x^{6}\\
&+(5\,\xi^4+4\,\xi^2+5\,\xi+1)\,x^{5}+(2\,\xi^4+5\,\xi^3+2\,\xi^2+4\,\xi+2)\,x^{4}\\
&+(4\,\xi^3+2\,\xi^2+5\,\xi+3)\,x^{3}+(3\,\xi^4+2\,\xi^3+2\,\xi^2+5\,\xi+2)\,x^{2}\\
&+(5\,\xi^4+5\,\xi^3+2\,\xi^2+2\,\xi+5)\,x+5\,\xi^4+\xi^3+3\,\xi^2+6\,\xi+6.
\end{aligned}
$$

L'équation de Weierstrass réduite d'une seconde courbe ℓ-isogène peut être obtenue de la même manière : $y^2=x^3+\tilde{a}_4''\,x+\tilde{a}_6''$, avec

$$
\begin{aligned}
\tilde{a}_4'' &= -20864088\,\xi^4-7284958\,\xi^3-98572779\,\xi^2+47355439\,\xi-119448910+O(7^{10}),\\
\tilde{a}_6'' &= -109290021\,\xi^4+30262957\,\xi^3+17886552\,\xi^2+61220134\,\xi+98475059+O(7^{10}).
\end{aligned}
$$

L'équation différentielle liée à cette seconde courbe a pour solution modulo $x^{4\ell}$:

$$
\begin{aligned}
&\left(38\,\xi^4+167\,\xi^3-156\,\xi^2-119\,\xi+45+O(7^3)\right)\,x^{187}+\left(O(7^4)\right)\,x^{186}\\
&+\left(-115\,\xi^4-141\,\xi^3+25\,\xi^2-135\,\xi+167+O(7^3)\right)\,x^{185}+\left(O(7^4)\right)\,x^{184}\\
&+\left(-81\,\xi^4+114\,\xi^3-158\,\xi^2-80\,\xi-10+O(7^3)\right)\,x^{183}+\left(O(7^4)\right)\,x^{182}\\
&+\left(-49\,\xi^4-126\,\xi^3-149\,\xi^2-30\,\xi-125+O(7^3)\right)\,x^{181}+\left(O(7^4)\right)\,x^{180}\\
&+\left(-26\,\xi^4-161\,\xi^3+5\,\xi^2-9\,\xi+150+O(7^3)\right)\,x^{179}+\left(O(7^4)\right)\,x^{178}\\
&+\left(99\,\xi^4+109\,\xi^3-33\,\xi^2+133\,\xi-7+O(7^3)\right)\,x^{177}+\left(O(7^4)\right)\,x^{176}\\
&+\ \ldots\\
&+\left(-16380744\,\xi^4-18740084\,\xi^3-16581200\,\xi^2-2329120\,\xi-16318206+O(7^9)\right)\,x^{11}\\
&+\left(-7045870\,\xi^4-1490874\,\xi^3-1892621\,\xi^2+15345025\,\xi-16524892+O(7^9)\right)\,x^{9}\\
&+\left(12370373\,\xi^4-18015164\,\xi^3-18899193\,\xi^2-15803937\,\xi-13142871+O(7^9)\right)\,x^{7}\\
&+\left(54408641\,\xi^4-85471071\,\xi^3-38104803\,\xi^2-136502081\,\xi+101045208+O(7^{10})\right)\,x^{5}\\
&+\left(1+O(7^{10})\right)\,x \mod x^{188}\,.
\end{aligned}
$$

Comme lors du calcul du premier polynôme minimal d'un point de ℓ-torsion, la précision 7-adique réellement nécessaire au calcul se limite à 8, puisque la précision absolue des termes de plus haut degré de la solution calculée est trop élevée de 2.

Le second polynôme minimal d'un point de ℓ-torsion calculé par l'algorithme se déduit de la solution de l'équation différentielle calculée :

$$
\begin{aligned}
&x^{23} && +(2\,\xi^4+6\,\xi^3+6\,\xi^2+5\,\xi+1)\,x^{22}\\
&+(6\,\xi^4+6\,\xi^3+5\,\xi^2+\xi+6)\,x^{21} && +(\xi^4+3\,\xi^2+\xi+3)\,x^{20}\\
&+(5\,\xi^4+3\,\xi^3+6\,\xi^2+5\,\xi+2)\,x^{19} && +(\xi^4+2\,\xi^3+4\,\xi+4)\,x^{18}\\
&+(5\,\xi^4+5\,\xi^3+2\,\xi^2+3\,\xi+5)\,x^{17} && +(2\,\xi^4+2\,\xi^3+4\,\xi^2+\xi+5)\,x^{16}\\
&+(5\,\xi^4+\xi^3+2\,\xi^2+\xi+2)\,x^{15} && +(4\,\xi^3+3\,\xi^2+2\,\xi+5)\,x^{14}\\
&+(4\,\xi^4+3\,\xi^2+5\,\xi+1)\,x^{13} && +(3\,\xi^4+3\,\xi^3+\xi^2+3\,\xi+6)\,x^{12}\\
&+(5\,\xi^4+\xi^3+3\,\xi^2+2)\,x^{11} && +(2\,\xi^4+3\,\xi^3+\xi^2+2\,\xi+1)\,x^{10}\\
&+(5\,\xi^4+\xi^3+3\,\xi^2+4\,\xi+1)\,x^{9} && +(3\,\xi^4+\xi^3+6\,\xi^2+4\,\xi+1)\,x^{8}\\
&+(6\,\xi^4+5\,\xi^3+3\,\xi^2+3)\,x^{7} && +(\xi^4+4\,\xi^3+4\,\xi^2+3\,\xi)\,x^{6}\\
&+(5\,\xi^2+2\,\xi+6)\,x^{5} && +(\xi^4+6\,\xi^3+5\,\xi^2+5\,\xi+5)\,x^{4}\\
&+(\xi^4+3\,\xi^3+2\,\xi+5)\,x^{3} && +(4\,\xi^4+5\,\xi^3+6\,\xi^2+4\,\xi+2)\,x^{2}\\
&+(4\,\xi^3+6\,\xi+4)\,x && +3\,\xi^4+1.
\end{aligned}
$$

1.5 Conclusion

Dans ce chapitre, on a présenté le contexte des courbes elliptiques, ainsi que la problématique du calcul des points de ℓ-torsion sur ces courbes. Par le biais d'un relèvement dans les p-adiques avec une précision seulement logarithmique en ℓ, on a montré comment résoudre en caractéristique quelconque l'équation différentielle correspondant au calcul des isogénies sous-jacentes à la ℓ-torsion. La résolution de cette équation différentielle représente $\tilde{O}(\ell \log(q))$ opérations élémentaires : le calcul des polynômes minimaux de deux points de ℓ-torsion nécessite donc $\tilde{O}(\ell \max(\ell, \log(q))^2)$ opérations élémentaires, en caractéristique quelconque.

Chapitre 2

Modèle générique des groupes avec couplage

Depuis quelques années, les groupes avec couplage sont considérés comme des blocs élémentaires standards afin de constituer des primitives cryptographiques. La sécurité de schémas basés sur de tels groupes, tout comme celle de nombreux schémas précédents n'utilisant pas les couplages, repose sur des hypothèses reliées au problème du logarithme discret. Comme ces hypothèses ne sont pas prouvées difficiles, il est souhaitable de disposer au moins d'arguments suggérant qu'elles sont difficiles. Pour cela, Nechaev [Nec93] et Shoup [Sho97] ont introduit le modèle du groupe générique. Dans ce modèle, qui se rapproche par certains aspects du modèle de l'oracle aléatoire introduit par Bellare et Rogaway [BR93], on peut prouver la difficulté du problème du logarithme discret, et d'hypothèses proches.

Au cours du temps, ce modèle a été généralisé dans différentes directions pour prendre en compte de nouvelles propriétés, ou s'adapter à des utilisations différentes des mêmes types de groupes. La pertinence de ce modèle est toutefois sujette à des critiques : en particulier, le fait que la réponse à toute requête fraîche soit simulée dans les preuves par une suite aléatoire contrevient à ce que l'on attend d'une loi de groupe usuelle.

Le but de ce chapitre consiste à présenter un modèle générique de groupes prenant en compte les diverses améliorations du modèle de Nechaev et Shoup, et notamment l'utilisation de couplages, en s'appuyant sur des définitions rigoureuses de familles de groupes. On peut dès lors présenter un cadre général permettant de prouver facilement la difficulté d'hypothèses « raisonnables » dans ce modèle, par le biais d'un lemme fondamental (lemme 2.1) et de deux exemples d'utilisation (problème bilinéaire Diffie-Hellman en section 2.4.3 et problème q-BDHI en section 2.4.4).

Ce modèle peut de plus être étendu en utilisant une notion de famille pseudo-aléatoire de groupes. Les résultats qui en découlent diffèrent relativement peu de ceux du modèle générique classique, mais ils s'appliquent à des familles de groupes un peu plus réalistes.

2.1 Introduction au modèle générique

Le problème du logarithme discret est un cadre général utilisé pour construire des fonctions à trappe dans des protocoles cryptographiques asymétriques. Il peut être utilisé à partir du moment où on sait construire des familles de groupes cycliques vérifiant les propriétés suivantes :

- la loi de groupe et l'inversion dans le groupe peuvent être calculés efficacement (et par conséquent, le calcul de l'exponentiation est également efficace en utilisant un algorithme de type « *Square and Multiply* ») ;
- le calcul d'un logarithme discret est difficile dans la famille de groupes.

On ne connaît cependant aucune famille concrète de groupes cycliques pour laquelle cette dernière propriété est formellement prouvée. Heureusement, d'un point de vue pratique, on ne connaît pas d'algorithme meilleur que les algorithmes génériques (comme l'algorithme Rho de Pollard [MvOV01]) pour calculer des logarithmes discrets dans certaines familles de groupes, comme par exemple la familles des points rationnels sur des courbes elliptiques définies sur un corps fini. Ces algorithmes génériques ont une complexité exponentielle en temps, et on peut donc croire en la difficulté du problème du logarithme discret sur cette famille de groupe.

Certains cryptosystèmes largement utilisés, comme par exemple l'échange de clés Diffie-Hellman et ses variantes [DH76, MTI86], le chiffrement El Gamal [Gam84, Gam85] ou DSA et les schémas de signatures proches [N06, Gam84, Gam85, Sch91], utilisent des fonctions à trappe dont la sécurité est basée sur la difficulté du logarithme discret.

2.1.1 Modèle du groupe générique

Cette situation n'est pas vraiment satisfaisante, et, à défaut d'une preuve complète et rigoureuse, on aimerait disposer d'un argument positif de sécurité du problème du logarithme discret, au lieu de constater l'absence actuelle d'algorithmes efficaces pour résoudre ce problème. Le modèle du groupe générique proposé par Nechaev et Shoup est destiné à donner cet argument positif de sécurité. Plus précisément, les articles [Nec93, Sho97] définissent la notion d'algorithmes génériques, qui sont des automates à mémoire capables de faire des opérations de groupe, à l'exclusion de toute autre opération. Dans ce modèle de calcul, les auteurs prouvent que les algorithmes les plus rapides pour résoudre le problème du logarithme discret et le problème Diffie-Hellman sont exponentiels en temps.

Dans une formulation moderne du modèle du groupe générique, les adversaires sont des machines de Turing qui manipulent des chaînes de bits au lieu d'éléments d'un groupe, et qui sont incapables de réaliser des opérations de groupe sur ces chaînes de bits. Dans les preuves, les lois de groupes sont fournies à ces machines par des oracles dont le fonctionnement est simulé de la manière suivante :

- toutes les requêtes à ces oracles sont enregistrées dans une liste ;
- quand une nouvelle requête est similaire à une requête précédente, les oracles renvoient la même réponse ;

– dans le cas contraire, c'est-à-dire pour une nouvelle requête fraîche, les oracles renvoient une chaîne de bits tirée aléatoirement.

À la fin du jeu, le challenger choisit aléatoirement une valeur du logarithme discret de chacune des entrées de la machine de Turing, et vérifie que la simulation réalisée avec les oracles est cohérente avec ces valeurs (les oracles ne doivent pas avoir répondu différemment à des requêtes correspondant à un même élément de groupe). On remarque la proximité avec le modèle de l'oracle aléatoire [BR93] où, de manière proche, une requête déjà posée aboutit à la même réponse alors qu'une requête fraîche a une réponse tirée aléatoirement. La différence essentielle est que dans le modèle de l'oracle aléatoire, il n'y a pas de structure de groupe, et ainsi aucune incompatibilité possible. De plus, le modèle de l'oracle aléatoire est souvent utilisé dans des preuves nécessitant la « programmation » de l'oracle, alors que cette technique n'existe pas dans le modèle du groupe générique.

Ce modèle du groupe générique a été par la suite généralisé pour prendre en compte de nouvelles propriétés des groupes utilisées dans les protocoles comme les couplages [BB04b, YW05]. Le modèle du groupe générique est ainsi devenu un outil standard pour prouver des protocoles cryptographiques. Il peut être utilisé directement pour évaluer le coût d'une attaque sur un protocole [Bro05, LS08b], ou plus généralement pour prouver la solidité d'une nouvelle hypothèse mathématique sur laquelle se base un nouveau protocole cryptographique [Jou00, BB04a, BB04b].

2.1.2 Limites et critiques

Comme le modèle du groupe générique est le seul argument positif utilisé pour justifier la robustesse d'hypothèses dont dépendent la sécurité de nombreux protocoles cryptographiques, il est légitime de vouloir évaluer la pertinence de ce modèle.

Dans un premier temps, comme mentionné dans [AF07], le modèle du groupe générique dans une formulation classique est loin de traiter le cas de machines de Turing générales ayant accès à des oracles. En effet, à partir d'une chaîne de bits, issue par exemple d'un oracle, une machine de Turing peut inverser un bit, et utiliser la chaîne résultante pour construire une nouvelle requête à un oracle, ce qui n'est manifestement pas prévu dans le modèle. Intuitivement, ce type d'opérations n'est pas en mesure d'aider un adversaire à améliorer significativement sa probabilité de casser la sécurité d'une hypothèse, mais on doit formellement prendre ces opérations en compte dans un modèle étendu.

Outre cette prise en compte nécessaire de certaines généralisations, le modèle du groupe générique a été la cible de nombreuses critiques [Den02, SPMLS02, NSS04], même si certaines de ces critiques ont été levées dans [KM07]. Ainsi, il est curieux que dans les simulations utilisées dans les preuves, les oracles pour lois de groupe répondent aux requêtes fraîches par des chaînes de bits tirées aléatoirement : ce comportement est extrêmement différent de l'utilisation de groupes concrets, comme expliqué dans [KM07].

2.1.3 Contribution

Dans ce chapitre, on présente dans un premier temps une formalisation complète du modèle du groupe générique, s'appuyant sur des définitions mathématiques précises de familles de groupes, avec ou sans couplage. Cette formalisation tient compte des diverses extensions réalisées sur le modèle générique, en particulier sur la capacité d'un adversaire à utiliser des chaînes de bits qui ne lui sont pas données lors de l'initialisation, et qui ne sont pas non plus des réponses données par les oracles.

De plus, l'approche suivie dans l'analyse d'un problème donné est plus naturelle que l'approche classique utilisée. Habituellement, comme c'est notamment le cas dans [BB04b, YW05], dans le modèle du groupe générique, les oracles sont simulés par un tirage aléatoire de chaînes de bits lors de requêtes fraîches, sans tenir compte d'une structure de groupe. Ce n'est qu'a posteriori qu'une structure de groupe est utilisée, pour déterminer si les réponses sont cohérentes ou si une collision a eu lieu dans les requêtes, de telle sorte que les oracles ont répondu différemment à des requêtes qui se révèlent être identiques.

À l'inverse, dans le modèle présenté ici, on tire aléatoirement une structure de groupe dès le début du jeu, et les réponses renvoyées par les oracles suivent cette structure. L'analyse s'appuie sur l'ensemble des structures de groupes qui sont indistinguables pour l'adversaire de la structure de groupe utilisée réellement. À partir de cet ensemble, on peut calculer une probabilité de réussite de l'adversaire. Cette seconde approche présente notamment l'avantage de définir clairement l'espace de probabilités utilisé.

Dans l'approche suivie, les réponses aux oracles sont donc calculées à partir d'une loi de groupe prédéterminée, et non plus choisies aléatoirement au cours du jeu. Cependant, la loi de groupe est déterminée initialement par un choix aléatoire parmi toutes les lois de groupe envisageables, ce qui reste peu réaliste. Pour améliorer cet aspect du modèle, on définit la notion de famille pseudo-aléatoire de groupes : dans une telle famille, les lois sont construites à partir d'une famille pseudo-aléatoire de permutations. Ces familles pseudo-aléatoires de groupes constituent une généralisation du modèle du groupe générique.

En fait, on peut prouver que tout résultat obtenu dans le modèle du groupe générique peut être transféré dans ce modèle basé sur des familles pseudo-aléatoires de groupes. Autrement dit, on peut considérer des familles de groupes où les lois de groupes sont tirées suivant une distribution arbitraire. La sécurité d'une hypothèse dans ce contexte se réduit à la solidité de la famille de fonctions sous-jacente en tant que famille de fonctions pseudo-aléatoire, et à la sécurité de cette même hypothèse dans le modèle du groupe générique. Ainsi, non seulement l'approche choisie pour aborder le modèle générique est formellement plus précise, mais elle permet une plus grande souplesse d'utilisation.

2.1.4 Organisation du chapitre

Dans les parties 2.2 et 2.3, on définit précisément les notions de famille générique de groupes cycliques, sans couplage puis avec couplage, par rapport à des familles

de représentations de groupes cycliques. Dans la partie 2.4, on présente des outils techniques permettant d'évaluer la sécurité d'une hypothèse dans le modèle du groupe générique. Dans la partie 2.5, le modèle est étendu pour prendre en compte les familles pseudo-aléatoires de groupes cycliques.

2.2 Groupes cycliques et leurs représentations

Pour simplifier les notations, toutes les définitions s'appuient sur des groupes notés additivement. En revanche, lorsqu'on mentionne des groupes concrets (courbes elliptiques, corps finis), on utilisera les notations usuelles.

2.2.1 Unicité du groupe cyclique d'ordre n

Pour tout entier $n \in \mathbb{N}^*$, il existe un unique groupe cyclique d'ordre n à isomorphisme près : deux groupes cycliques sont isomorphes si et seulement s'ils ont le même ordre. En particulier, un groupe cyclique G est d'ordre n si et seulement s'il est isomorphe au groupe additif $\mathbb{Z}/n\mathbb{Z}$.

Il existe naturellement plusieurs isomorphismes reliant G à $\mathbb{Z}/n\mathbb{Z}$. Mais lorsque g est un générateur du groupe G, on considère spécifiquement l'isomorphisme de groupes qui envoie $g \in G$ sur $1 \in \mathbb{Z}/n\mathbb{Z}$: il s'agit simplement du logarithme discret sur G en base g, noté $\log_g$.

2.2.2 Structure de groupe héritée d'une bijection

Soit A un ensemble fini de n éléments. Toute bijection f, de $\mathbb{Z}/n\mathbb{Z}$ vers A, munit A d'une structure de groupe. Les lois de groupes sur A sont notées $+_f$ et $-_f$ et sont simplement dérivées de la structure de groupe additif de $\mathbb{Z}/n\mathbb{Z}$: pour sommer deux éléments sur A, on prend leurs images réciproques par f dans $\mathbb{Z}/n\mathbb{Z}$, on réalise l'addition, puis on revient dans A par la bijection f (le principe est le même pour l'opposé). Ainsi, les lois sont définies par les règles suivantes :

$$\forall (x, y) \in A^2,\ x +_f y = f(f^{-1}(x) + f^{-1}(y)) \text{ and } -_f x = f(-f^{-1}(x)). \tag{2.1}$$

On note A_f l'ensemble A muni de cette structure de groupe. On remarque que, par construction, $f(0)$ est l'élément neutre de A_f, tandis que $f(1)$ est l'un de ses générateurs.

2.2.3 Famille générique de groupes cycliques

On considère l'ensemble $\mathcal{F}(A)$ de toutes les bijections de $\mathbb{Z}/n\mathbb{Z}$ dans un ensemble A de n éléments. Pour tout sous-ensemble $S \subset \mathcal{F}(A)$, on définit la famille $A_S = \{A_f, f \in S\}$. Cette famille est un ensemble de structures de groupe cyclique sur l'ensemble A. La famille de groupes cycliques qui correspond au modèle du groupe générique donné dans [Sho97] est appelée famille générique de groupes cycliques et définie par :

Définition 2.1 (Famille générique de groupes cycliques) *Soit $B(n)$ l'ensemble des écritures binaires des entiers entre 0 et $n-1$. La famille $B(n)_{\mathcal{F}(B(n))}$ est appelée famille générique de groupes cycliques d'ordre n. L'union de toutes ces familles, lorsque n est un entier non-nul, est appelée famille générique de groupes cycliques.*

La famille générique de groupes cycliques contient ainsi des groupes de toute taille n. Pour un entier n donné, elle contient toutes les structures de groupes possibles sur $B(n)$. Lors d'un jeu sur une hypothèse donnée, un adversaire aura accès à des oracles réalisant des opérations de groupe, sur une structure fixée mais inconnue de lui.

Lorsque la structure de groupe est définie par une bijection $f : \mathbb{Z}/n\mathbb{Z} \to B(n)$, un adversaire peut obtenir l'image par f de n'importe quel élément de $\mathbb{Z}/n\mathbb{Z}$ à partir du générateur $f(1)$ en au plus $2\log_2(n)$ additions, via l'algorithme *Square and Multiply*. L'accès aux oracles permet donc de calculer des images de la fonction f en temps polynomial.

À l'inverse, on veut s'assurer que l'adversaire n'est pas en mesure de calculer lui-même les opérations de groupe, sans faire appel aux oracles. En fait, le seul moyen pour l'adversaire de réaliser des opérations de groupe est d'utiliser les règles (2.1) qui nécessitent le calcul de f^{-1}. Or, cette application f^{-1} est l'isomorphisme canonique mentionné en section 2.2.1, associé au groupe $B(n)$ pour le générateur $f(1)$, c'est-à-dire le logarithme discret en base $f(1)$. Il est par conséquent difficile pour l'adversaire de calculer lui-même des opérations de groupe, sans passer par les oracles.

2.2.4 Famille de représentations de groupes cycliques

On doit désormais présenter un formalisme correspondant à une famille réelle de groupes, telle que celles qui sont utilisées concrètement dans les algorithmes et protocoles cryptographiques. Pour cela, on donne la définition suivante :

Définition 2.2 (Famille de représentations de groupes cycliques) *Soit L un langage sur $\{0,1\}$, c'est-à-dire un sous-ensemble de $\{0,1\}^*$. Une famille de représentations de groupes cycliques sur L est :*

- *la donnée d'un ensemble infini Ω de chaînes binaires, et d'une fonction $c : \Omega \to \mathbb{N}^*$ calculable en temps polynomial, tels que $\forall N \in \mathbb{N}, \exists\, \alpha \in \Omega \,/\, c(\alpha) \geqslant N$;*
- *pour tout paramètre $\alpha \in \Omega$, la donnée d'un sous ensemble fini L_α de L, de taille $c(\alpha)$, de deux lois sur L_α calculables en temps polynomial, $+_\alpha$ et $-_\alpha$, et deux éléments de L_α, 0_α et g_α. On impose que L_α muni des lois $+_\alpha$ et $-_\alpha$ soit un groupe cyclique d'élément neutre 0_α et de générateur g_α. On requiert également que $\max\{|x|, x \in L_\alpha\} = O(\log(c(\alpha)))$.*

Dans cette définition, à chaque paramètre correspond un ensemble, muni de la description de deux lois de groupe, et de l'élément neutre et d'un générateur, ce qui semble raisonnable. La taille du groupe est connue à partir du paramètre, via l'application c. On justifie désormais les différentes contraintes mentionnées dans cette définition.

On exige dans un premier temps que l'ensemble des paramètres soit infini, et que $(c(\alpha))_{\alpha\in\Omega}$ soit non-borné. Ces contraintes sont nécessaires pour pouvoir étudier le comportement asymptotique d'un adversaire sur cette famille de représentations de groupes cycliques : il est indispensable de pouvoir faire tendre la taille des groupes de la famille vers l'infini.

Ensuite, on demande une représentation efficace des éléments du groupe dans L. C'est une exigence liée à la complexité du calcul. Les lois sont calculables en temps polynomial en la taille de leurs entrées, mais elles doivent être calculables polynomialement en le paramètre global de complexité, c'est-à-dire $\log(c(\alpha))$. Demander une représentation efficace des éléments du groupe est une contrainte faible qui résout ce problème.

Exemple 2.1 *On considère par exemple la famille des courbes elliptiques d'ordre premier définies sur des corps finis $\mathbb{F}_p$ où p est un nombre premier supérieur ou égal à 5. Une courbe elliptique E de cette famille peut être représentée par un 7-uplet $(p, a_1, a_2, a_3, a_4, a_6, g)$ définissant son équation de Weierstrass $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$ sur $\mathbb{F}_p$ et l'un de ses générateurs g.*
La fonction c calculant $\#E(\mathbb{F}_p)$ à partir d'un paramètre $\alpha = (p, a_1, a_2, a_3, a_4, a_6, g)$ est basée sur l'algorithme polynomial dû à Schoof. Grâce au théorème de Hasse, on sait que $|\#E(\mathbb{F}_p) - p - 1| \leqslant 2\sqrt{p}$. Par ailleurs, tous les entiers compris entre $p - 1 - 2\sqrt{p}$ et $p - 1 + 2\sqrt{p}$ sont cardinaux d'une courbe elliptique sur $\mathbb{F}_p$: il existe en particulier une courbe d'ordre p ! On a ainsi la certitude de trouver des courbes de taille arbitrairement grande dans cette famille : il suffit de prendre de grandes valeurs de p.
Un élément de $E(\mathbb{F}_p)$ peut être représenté par un couple $(x, y) \in (\mathbb{F}_p)^2$ vérifiant l'équation de Weierstrass. Les lois de groupe, le générateur g et l'élément neutre de $E(\mathbb{F}_p)$ sont directement dérivés du paramètre $\alpha = (p, a_1, a_2, a_3, a_4, a_6, g)$.
On déduit de ces propriétés que l'ensemble de paramètres $(p, a_1, a_2, a_3, a_4, a_6, g)$ et l'ensemble des groupes cycliques $\{(x, y) = \lambda\, g \,/\, \lambda \in \mathbb{N}\}$ constituent bien une famille de représentations de groupes cycliques.

Exemple 2.2 *D'après la section 2.2.3, un élément de la famille générique de groupes cycliques est défini par un ordre n, et deux lois, $+_f$ et $-_f$. Pour relier cette famille avec la définition de famille de représentations de groupes cycliques, on utilise la présentation suivante :*

- *le paramètre associé à un élément de la famille générique de groupes cycliques est un entier n, avec deux éléments de $B(n)$ correspondant à 0_f et g_f ; la fonction $c : (n, 0_f, g_f) \mapsto n$ permettant le calcul de la taille du groupe est alors triviale ;*
- *l'ensemble des éléments du groupe associé au paramètre $(n, 0_f, g_f)$ est $B(n)$,*
- *deux oracles, $+_f$ et $-_f$, sont construits (voir section 2.2.2) à partir d'une fonction f aléatoirement choisie dans $\mathcal{F}(B(n))$ et vérifiant $f(0) = 0_f$ et $f(1) = g_f$.*

La différence avec la définition formelle d'une famille de représentations de groupes cycliques est que les lois de groupe ne peuvent pas être déduites du paramètre. Ces lois de groupes ne sont fournies que sous la forme d'oracles.

2.2.5 Quelques problèmes standards

Une fois ces notions de familles établies, on peut définir quelques problèmes standards pour des familles de représentations de groupes cycliques :

Définition 2.3 *Soit $(\Omega, (L_\alpha)_{\alpha\in\Omega})$ une famille de représentations de groupes cycliques sur un langage L. Dans cette famille,*
- *un algorithme résolvant le problème du logarithme discret calcule $\log_{g_\alpha}(x)$ dans le groupe L_α à partir des données $\alpha \in \Omega$, $x \in L_\alpha$;*
- *un algorithme résolvant le problème Diffie-Hellman calcule $\log_{g_\alpha}(x)\,.\,y$ dans le groupe L_α à partir des données $\alpha \in \Omega$, $(x, y) \in (L_\alpha)^2$;*
- *un algorithme résolvant le problème décisionnel Diffie-Hellman décide si z est égal à $\log_{g_\alpha}(x)\,.\,y$ dans le groupe L_α à partir des données $\alpha \in \Omega$, $(x, y, z) \in (L_\alpha)^3$.*

En suivant l'exemple 2.2, on peut étendre ces définitions au cas de la famille générique de groupes cycliques. Un algorithme résolvant l'un de ces problèmes a exactement les mêmes entrées et sorties : sa seule limitation est qu'il doit utiliser des oracles au lieu de pouvoir calculer directement les lois de groupe.

2.3 Groupes cycliques avec couplage

2.3.1 Couplages parfaits

Soit G un groupe cyclique d'ordre n. Un tel groupe a une structure canonique de $\mathbb{Z}$-module : $\forall (k, x) \in \mathbb{N} \times G,\ k.x = x + x + \ldots + x$ et $(-k).x = -(k.x)$. On note $\hat{G}$ le dual de ce groupe, c'est-à-dire le groupe de $(\mathbb{Z}/n\mathbb{Z})$-formes linéaires sur G.

Un couplage est une application $(\mathbb{Z}/n\mathbb{Z})$-bilinéaire d'un couple de groupes cycliques dans un troisième groupe cyclique, tous de même ordre n. Il existe un couplage de $G \times \hat{G}$ dans $\mathbb{Z}/n\mathbb{Z}$, appelé couplage canonique de G et défini par :

$$e_c : \left(\begin{array}{ccc} G \times \hat{G} & \to & \mathbb{Z}/n\mathbb{Z} \\ (x, v) & \mapsto & v(x) \end{array} \right)$$

Soient G, G' et G'' trois groupes cyclique de même ordre n. Un couplage e de $G \times G'$ dans G'' est dit parfait s'il est isomorphe (en tant que couplage) au couplage canonique de G, c'est-à-dire s'il existe trois isomorphismes de groupes $m : G \to G$, $m' : G' \to \hat{G}$ et $m'' : \mathbb{Z}/n\mathbb{Z} \to G''$ tels que $\forall (x, y) \in G \times G'$, $e(x, y) = m''(e_c(m(x), m'(y)))$.

$$\begin{array}{ccc} G \times G' & \xrightarrow{\ e\ } & G'' \\ {\scriptstyle m}\downarrow \quad \downarrow{\scriptstyle m'} & & \uparrow{\scriptstyle m''} \\ G \times \hat{G} & \xrightarrow[\ e_c\ ]{} & \mathbb{Z}/n\mathbb{Z} \end{array}$$

On considère un ensemble A de n élément, et un couple $(f, h) \in \mathcal{F}(A)^2$: on souhaite décrire l'ensemble des couplages parfaits de $A_f \times A_f$ dans A_h. Par définition, un tel couplage se déduit de trois isomorphismes de groupes m, m' et m'' :

$$\begin{array}{ccc} A_f \times A_f & \xrightarrow{e} & A_h \\ {\scriptstyle m}\downarrow & \downarrow{\scriptstyle m'} & \uparrow{\scriptstyle m''} \\ A_f \times \widehat{A_f} & \xrightarrow[e_c]{} & \mathbb{Z}/n\mathbb{Z} \end{array}$$

Sans perte de généralité, on peut fixer $m = id$, $m'' = h$ et $m' = \hat{f}^{-1} \circ i \circ f^{-1}$, où i est un isomorphisme quelconque de $\mathbb{Z}/n\mathbb{Z}$ dans $\widehat{\mathbb{Z}/n\mathbb{Z}}$, et où $\hat{f}$ est l'isomorphisme dual de f, c'est-à-dire $\hat{f} : v \in \widehat{A_f} \mapsto v \circ f \in \widehat{\mathbb{Z}/n\mathbb{Z}}$.

Pour tout ensemble A et tout couple $(f, h) \in \mathcal{F}(A)^2$, la description d'un couplage parfait de $A_f \times A_f$ dans A_h est alors équivalente à la donnée d'un isomorphisme i de $\mathbb{Z}/n\mathbb{Z}$ dans $\widehat{\mathbb{Z}/n\mathbb{Z}}$.

Remarque 2.1 *Le couplage parfait e, en tant qu'application bilinéaire, est défini de manière unique par la valeur de $e(f(1), f(1))$. De même l'isomorphisme i est défini de manière unique par la valeur de $i(1)$, et indirectement par la valeur de $e(f(1), f(1))$. On en déduit que pour tout couple $(f, h) \in \mathcal{F}(A)^2$, il existe un unique couplage parfait e tel que $e(f(1), f(1)) = h(1)$.*

Dans la suite de ce chapitre, on limitera l'étude des groupes avec couplage au seul cas du couplage parfait vérifiant $e(f(1), f(1)) = h(1)$. Comme le couplage défini par $(x, y) \in {A_f}^2 \mapsto h(f^{-1}(x) \,.\, f^{-1}(y)) \in A_h$ convient, c'est l'unique couplage parfait mentionné dans la remarque précédente.

2.3.2 Famille générique des groupes cycliques avec couplage

On suit une démarche similaire à celle de la section 2.2.3 : on considère un ensemble A de n éléments, et l'ensemble $\mathcal{F}(A)$ de toutes les bijections de $\mathbb{Z}/n\mathbb{Z}$ dans A. Pour tout sous-ensemble $S \subset \mathcal{F}(A)$, soit $\mathcal{P}(A, S)$ la famille de groupes cycliques avec couplage paramétrée par l'ensemble $\{(f, h) \in S^2\}$. À partir d'un couple $(f, h) \in S^2$, on construit les structures de groupes A_f et A_h en suivant l'équation (2.1). On construit de plus le couplage parfait $e_{f,h}$ de $A_f \times A_f$ dans A_h défini par :

$$e_{f,h} : \left(\begin{array}{ccc} A_f \times A_f & \to & A_h \\ (x, y) & \mapsto & h\big(f^{-1}(x) \,.\, f^{-1}(y)\big) \end{array} \right)$$

Définition 2.4 (Famille générique de groupes cycliques avec couplage) *Soit $B(n)$ l'ensemble des écritures binaires des entiers entre 0 et $n-1$. La famille de groupes cycliques avec couplage $\mathcal{P}(B(n), \mathcal{F}(B(n)))$ est appelée famille générique de groupes cycliques d'ordre n avec couplage. L'union de toutes ces familles, lorsque n est un entier non-nul, est appelée famille générique de groupes cycliques avec couplage.*

Comme expliqué précédemment, les morphismes f et h peuvent être efficacement calculés à partir des structures de groupes de ${B(n)}_f$ et de ${B(n)}_h$. À l'inverse, les applications réciproques f^{-1} et h^{-1} ne sont pas directement accessibles.

2.3.3 Famille de représentations de groupes cycliques avec couplage

Définition 2.5 *Soient L et M deux langages sur $\{0,1\}$. Une famille de représentations de groupes cycliques avec couplage, sur ces langages L et M, est :*

- *la donnée de deux familles de représentations de groupes cycliques, la première sur L et la seconde sur M : $(\Gamma, (L_\gamma)_{\gamma\in\Gamma})$ et $(\Delta, (M_\delta)_{\delta\in\Delta})$;*
- *la donnée d'un ensemble de paramètres $\Omega \subset \Gamma \times \Delta$;*
- *pour tout paramètre $\alpha = (\gamma, \delta) \in \Omega$, la donnée d'un couplage parfait e_α de $L_\gamma \times L_\gamma$ dans M_δ calculable en temps polynomial et tel que $e_\alpha(g_\gamma, g_\gamma) = g_\delta$.*

Lorsque Ω_1 (respectivement Ω_2) représente l'ensemble des parties gauches (respectivement droites) des éléments de l'ensemble des paramètres Ω, cette famille est notée :

$$\big(\Omega, (L_\gamma)_{\gamma\in\Omega_1}, (M_\delta)_{\delta\in\Omega_2}, (e_\alpha)_{\alpha\in\Omega}\big).$$

Exemple 2.3 *On poursuit le cas des courbes elliptiques présenté dans l'exemple 2.1. On considère l'ensemble des courbes elliptiques E définies sur un corps fini $\mathbb{F}_q$ de caractéristique p. Soit $l \geqslant 2$ un entier premier avec p, soit k le plus petit entier tel que l divise $q^k - 1$. Alors $\mathbb{F}_{q^k}$ est une extension de $\mathbb{F}_q$ qui contient les racine $l^{ème}$ de l'unité, et on peut considérer le couplage de Weil $e_W : E[l] \times E[l] \to \mathbb{F}_{q^k}$ (voir [Sil86]).*
Le couplage de Weil n'est cependant pas satisfaisant : il s'agit d'une forme bilinéaire antisymétrique sur $E[l]$. On doit alors considérer un sous-groupe isotrope maximal $G(E[l])$ de $E[l]$ et un isomorphisme avec son dual pour obtenir un couplage bilinéaire non-trivial $\tilde{e} : G(E[l]) \times G(E[l]) \to \mathbb{F}_{q^k}$, tel que si P est un générateur de $G(E[l])$ alors $\tilde{e}(P, P)$ est un générateur du groupe des racines $l^{ème}$ de l'unité dans $\mathbb{F}_{q^k}$.
Ce couplage parfait peut être calculé en temps polynomial probabiliste grâce à l'algorithme de Miller [CFA+05]. Ainsi, on obtient une famille de représentations de groupes cycliques avec couplage.

Exemple 2.4 *On a déjà remarqué dans l'exemple 2.2 que la famille générique de groupes cycliques peut être assimilée à une famille de représentations de groupes cycliques sur $\{0,1\}^*$. De même, la famille générique de groupes cycliques avec couplage peut être vue comme une famille de représentations de groupes cycliques avec couplage sur les langages $\{0,1\}^*$ et $\{0,1\}^*$.*
Pour cela, on utilise deux instances de la famille générique de groupes cycliques comme une famille de représentations de groupes cycliques. L'ensemble de paramètres est défini par l'ensemble des couples $((n, 0_f, g_f), (n, 0_h, g_h))$ où n est dans $\mathbb{N}^*$ (la seule limitation est que l'ordre soit le même). Comme pour les lois de groupes, le couplage est donné par le biais d'un oracle, calculé à partir des fonctions f et h définissant les structures de groupes, comme expliqué en section 2.3.2 : $(x, y) \mapsto h(f^{-1}(x) \,.\, f^{-1}(y))$.

2.3.4 Problèmes bilinéaires Diffie-Hellman

On peut désormais présenter quelques problèmes bilinéaires classiques dans le formalisme établi pour les familles de représentations de groupes cycliques avec couplage :

Définition 2.6 *Soit* $(\Omega, (L_\gamma)_{\gamma\in\Omega_1}, (M_\delta)_{\delta\in\Omega_2}, (e_\alpha)_{\alpha\in\Omega})$ *une famille de représentations de groupes cycliques avec couplage sur les langages* L *et* M*. Dans cette famille,*

- *un algorithme résolvant le problème bilinéaire Diffie-Hellman calcule l'élément* $\log_{g_\gamma}(w) \,.\, e_{(\gamma,\delta)}(x,y)$ *dans le groupe* M_δ *à partir des données* $(\gamma,\delta) \in \Omega$, $(w,x,y) \in (L_\gamma)^3$ *;*
- *un algorithme résolvant le problème bilinéaire décisionnel Diffie-Hellman décide si* z *est égal à* $\log_{g_\gamma}(w) \,.\, e_{(\gamma,\delta)}(x,y)$ *dans le groupe* M_δ *à partir des données* $(\gamma,\delta) \in \Omega$, $(w,x,y) \in (L_\gamma)^3$, $z \in M_\delta$*.*

Comme précédemment, on peut généraliser ces définitions au cas de la famille générique de groupes cycliques avec couplage, avec les mêmes entrées et sorties, mais avec seulement un accès à des oracles pour le calcul des différentes lois et du couplage, comme expliqué dans l'exemple 2.4.

À partir de maintenant, on utilise le modèle de calcul des machines de Turing ayant accès à des oracles (voir par exemple [Pap94]) pour étudier précisément les complexités en temps et en espace. L'unité de temps utilisée correspond au coût d'un appel à un oracle.

On remarque immédiatement qu'il existe une réduction polynomiale du problème bilinéaire Diffie-Hellman sur le problème du logarithme discret dans L_γ, à partir de l'algorithme *Square and Multiply*. Par conséquent, le problème bilinéaire Diffie-Hellman peut être résolu avec une complexité de l'ordre de $\sqrt{p}$, où p est le plus grand facteur premier de $\#L_\gamma$, en utilisant l'algorithme Rho de Pollard [Pol78] et la méthode proposée par Pohlig et Hellman dans [PH78]. Le but de la suite de ce chapitre est de montrer notamment qu'il n'existe pas d'algorithme ayant une meilleure complexité pour résoudre le problème bilinéaire Diffie-Hellman dans la famille générique de groupes cycliques avec couplage.

2.4 Analyse de complexité

Dans cette section, on présente des définitions et un lemme fondamental pour permettre de quantifier la difficulté d'un problème quelconque dans la famille générique de groupes cycliques avec couplage. La difficulté des problèmes bilinéaire Diffie-Hellman et q-BDHI est alors évaluée dans cette famille générique, en illustrations d'une méthode générale qui peut être utilisée sur une grande variété de problèmes.

Pour cela, on considère dans un premier temps un problème de manière générale. Une instance de ce problème est définie par la taille des groupes, et par un ensemble d'éléments de ces groupes : ces éléments sont les données du problème, et un algorithme résolvant le problème sur cette instance doit calculer une solution correspondant à ces données.

Plus qu'un algorithme, on étudie en fait une suite quelconque d'appels aux oracles. Lors d'une telle suite, on s'intéresse à la probabilité d'occurrence d'une collision. Il y a collision lorsque deux éléments d'un groupe, a priori distincts, se révèlent être égaux.

Pour étudier ces collisions, il est donc nécessaire de mémoriser les éléments des groupes correspondant aux réponses des oracles, et leurs modes de calcul (sous la forme de fractions rationnelles).

Dans une suite d'appels aux oracles, lorsqu'une collision apparaît, elle implique une structure particulière des groupes. La mise en évidence d'une telle structure peut notamment permettre d'identifier le logarithme discret d'une des données du problème, et la solution du problème s'en déduit souvent facilement. Par conséquent, dès qu'une collision apparaît, on considère que la suite d'appels aux oracles permet de résoudre le problème : les éventuels appels postérieurs à cette collision ne présentent alors plus d'intérêt.

En revanche, lorsque la suite d'appels aux oracles n'aboutit à aucune collision, aucune structure particulière des groupes ne peut être décelée. La réponse au problème est alors difficilement identifiable.

2.4.1 Définitions et stratégie d'analyse

Une instance d'un problème quelconque dans la famille générique de groupes cycliques avec couplage est décrite par les entrées suivantes :

- la taille des groupes, $n \in \mathbb{N}^*$;
- les éléments neutres et des générateurs des deux groupes, $(0_f, g_f, 0_h, g_h) \in (B(n))^4$;
- les éléments définissant l'instance du problème, sous la forme d'éléments de $B(n)$: un r-uplet $(x_1, \ldots, x_r) \in (B(n))^r$ et un s-uplet $(y_1, \ldots, y_s) \in (B(n))^s$.

Des oracles calculant les lois de groupes, $+_f$ et $+_h$, les lois inverses, $-_f$ et $-_h$, et le couplage $e_{f,g}$ sont construits à partir de deux bijections f et h, choisies aléatoirement dans $\mathcal{F}(B(n))$. Ces bijections f and h doivent envoyer 0 et 1 vers les éléments neutres et générateurs des groupes : $f(0) = 0_f$, $f(1) = g_f$, $h(0) = 0_h$, $h(1) = g_h$.

On considère une suite de requêtes à ces oracles. On s'intéresse à la probabilité d'occurrence d'une collision suite à ces requêtes, probabilité calculée sur l'ensemble des couples $(f, h) \in \mathcal{F}(B(n))^2$. Pour cela, on construit deux suites de listes, R et S, au fur et à mesure des requêtes. Les listes constituant ces deux suites sont à valeurs dans $B(n) \times (\mathbb{Z}/n\mathbb{Z})(X_1, \ldots, X_n, Y_1, \ldots, Y_n)$ où $(\mathbb{Z}/n\mathbb{Z})(X_1, \ldots, Y_n)$ est le corps des fractions rationnelles en les variables $X_1, \ldots, X_n$, $Y_1, \ldots, Y_n$.

Le premier élément d'un couple d'une liste d'une suite est un élément d'un groupe ($B(n)_f$ pour R, $B(n)_h$ pour S), le second élément est une fraction rationnelle définissant comment l'élément de groupe a été calculé. De cette manière, il y a collision s'il existe deux couples, dans une même liste, ayant le même élément de groupe, mais des fractions rationnelles distinctes : un même élément d'un groupe a été obtenu en utilisant deux méthodes de calcul différentes.

Au niveau des fractions rationnelles, chacune des inconnues $X_1, \ldots, X_n$ représente le logarithme discret d'un élément de $B(n)_f$, dans la base g_f. Il en va de même pour les inconnues $Y_1, \ldots, Y_n$ vis-à-vis du groupe $B(n)_h$ et de la base g_h. Des éléments de ces groupes dont les logarithmes discrets sont corrélés utilisent les mêmes inconnues.

Nominalement, les seules inconnues utilisées sont liées aux logarithmes discrets des éléments $x_1, \ldots, x_r, y_1, \ldots, y_s$ donnés dans l'instance du problème. Mais lorsqu'une requête à l'un des oracles utilise un élément de groupe qui n'est jamais apparu auparavant, cet élément est associé à une nouvelle inconnue, son logarithme discret étant complètement décorrélé des précédents. On utilise des fractions rationnelles, et pas uniquement des polynômes, car dans certains problèmes, les logarithmes discrets des entrées et de la solution sont corrélés via des fractions (c'est le cas du problème q-BDHI avec une solution dont le logarithme discret est un inverse).

Les listes R_k et S_k correspondent à l'état après les k premières requêtes aux oracles. Les entiers r_k et s_k indiquent le nombre de variables utilisées dans les listes R_k et S_k (r_k pour les variables de type X, s_k pour les variables de type Y). Enfin, on appelle ρ_k et σ_k les tailles respectives de R_k et de S_k.

Les listes R_0 et S_0, ainsi que les valeurs r_0, s_0, ρ_0 et σ_0 dépendent fortement des entrées du problème (voir les sections 2.4.3 et 2.4.4 pour des exemples).

En revanche, l'évolution de ces éléments dépend uniquement des requêtes aux oracles et des réponses à ces requêtes. Ainsi, lors du $(k+1)^{\text{ème}}$ appel à un oracle, on calcule les valeurs de r_{k+1}, s_{k+1}, ρ_{k+1}, σ_{k+1}, R_{k+1} et de S_{k+1} en les initialisant avec les valeurs de r_k, s_k, ρ_k, σ_k, R_k et S_k puis en les modifiant en fonction des cas.

- Nouvelles variables : lors d'un appel à un oracle $+_f$, $-_f$ ou $e_{f,h}$, pour chaque entrée $a \in B(n)$ de cet appel, lorsque a n'apparaît pas comme premier élément d'un couple dans R_{k+1}, la valeur de r_{k+1} est incrémentée de 1, on définit $x_{r_{k+1}} = a$ et on ajoute le couple $(x_{r_{k+1}}, X_{r_{k+1}})$ dans la liste R_{k+1}. On procède de même lors de l'utilisation de nouvelles valeurs dans un appel à un oracle $+_h$ ou $-_h$, en incrémentant s_{k+1}, en définissant $y_{s_{k+1}}$ et en ajoutant le couple $(y_{s_{k+1}}, Y_{s_{k+1}})$ dans la liste S_{k+1}.
- Fraction rationnelle associée : lors d'un appel à un oracle $+_f$, $-_f$ ou $e_{f,h}$, on recherche les fractions rationnelles associées aux entrées dans la liste R_{k+1}. Lors d'un appel à un oracle $+_h$ ou $-_h$, on recherche ces fractions dans la liste S_{k+1}. La fraction rationnelle associée à la requête à l'oracle est l'opposé de la fraction rationnelle recherchée (cas des oracles $-_f$ et $-_h$), la somme des deux fractions rationnelles recherchées (cas des oracles $+_f$ et $+_h$) ou leur produit (cas de l'oracle $e_{f,h}$).
- Insertion d'une réponse fraîche : lors d'un appel à un oracle $+_f$ ou $-_f$, si la fraction rationnelle associée à la requête n'apparaît pas dans un couple de R_{k+1}, alors on ajoute dans cette liste la réponse donnée par l'oracle avec la fraction rationnelle associée à la requête. De même, lors d'un appel à un oracle $+_h$, $-_h$ ou $e_{f,h}$, si la fraction rationnelle associée à la requête n'apparaît pas dans un couple de S_{k+1}, alors on ajoute dans S_{k+1} la réponse donnée par l'oracle avec la fraction rationnelle associée à la requête.

Remarque 2.2 *Dans cette mise à jour, au plus trois couples ont été ajoutés dans les listes R_{k+1} et S_{k+1} par rapport aux listes R_k et S_k (deux ajouts potentiels pour des*

variables nouvelles, en plus de celui pour la réponse de l'oracle). On a ainsi l'inégalité suivante : $\rho_{k+1} + \sigma_{k+1} \leqslant \rho_k + \sigma_k + 3$.

Les suites de listes R et S étant construites, on peut désormais définir formellement la notion de collision :

Définition 2.7 (Collision) *On dit qu'il y a collision dans les listes* $(\mathsf{R}_k, \mathsf{S}_k)$ *s'il existe deux couples* $((v_1, P_1)$ *et* $(v_2, P_2))$*, tous deux dans* R_k *ou tous deux dans* S_k*, tels que :* $v_1 = v_2$ *et* $P_1 \neq P_2$.

Comme mentionné précédemment, pour une suite d'appels aux oracles, on souhaite calculer la probabilité d'occurrence d'une collision, cette probabilité portant sur le choix des bijections f et h. On doit donc définir les choix possibles pour f et h.

Les listes R_k et S_k contiennent toutes les valeurs utilisées dans les appels aux oracles, et toutes les réponses faites par les oracles, avec les fractions rationnelles qui leur sont associées. Ainsi, tout couple de bijections pour lequel les appels aux oracles auraient conduit aux mêmes réponses sont structurellement indistinguables du couple réellement utilisé dans la construction des oracles. On caractérise ces couples avec la notion de compatibilité :

Définition 2.8 (Compatibilité) *Un couple de bijections* $(f, h) \in \mathcal{F}(B(n))^2$ *est dit compatible avec les listes* $(\mathsf{R}_k, \mathsf{S}_k)$ *si :*

- $\forall (v_R, P_R) \in \mathsf{R}_k$, $P_R(f^{-1}(x_1), \ldots, f^{-1}(x_{r_k}), h^{-1}(y_1), \ldots, h^{-1}(y_{s_k}))$ *est défini et est égal à* $f^{-1}(v_R)$ *;*
- $\forall (v_S, P_S) \in \mathsf{S}_k$, $P_S(f^{-1}(x_1), \ldots, f^{-1}(x_{r_k}), h^{-1}(y_1), \ldots, h^{-1}(y_{s_k}))$ *est défini et est égal à* $h^{-1}(v_S)$.

Après k appels aux oracles, le couple de bijections utilisé pour construire les oracles est nécessairement compatible avec les listes $(\mathsf{R}_k, \mathsf{S}_k)$. Ainsi, l'étude de probabilité réalisée par la suite va porter sur l'ensemble des couples de bijections compatibles avec ces listes.

Il reste à introduire la notion de cohérence, portant sur les images réciproques des éléments x_i et y_j par les fonctions f et h, fondamentale dans l'étude des collisions, puisque ce sont ces images réciproques qui déterminent l'apparition des collisions dans le jeu :

Définition 2.9 (Cohérence) *Un* $(r+s)$*-uplet* $(\alpha_1, \ldots, \alpha_r, \beta_1, \ldots, \beta_s) \in (\mathbb{Z}/n\mathbb{Z})^{r+s}$ *est dit cohérent avec un ensemble* Π *de fractions rationnelles dans* $(\mathbb{Z}/n\mathbb{Z})(X_1, \ldots, X_r, Y_1, \ldots, Y_s)$ *si :*

- $\forall P \in \Pi$, $P(\alpha_1, \ldots, \alpha_r, \beta_1, \ldots, \beta_s)$ *est défini ;*
- $\forall (P_1, P_2) \in \Pi^2, \big(P_1 \neq P_2 \Longrightarrow P_1(\alpha_1, \ldots, \beta_s) \neq P_2(\alpha_1, \ldots, \beta_s)\big)$.

Pour préciser le contenu de toutes ces notations les unes par rapport aux autres, et en vue d'introduire la stratégie de la preuve suivante, on imagine le jeu suivant : la situation après les k premiers appels aux oracles est modélisée par les listes $(\mathsf{R}_k, \mathsf{S}_k)$. On suppose que ces listes sont sans collision (voir définition 2.7) : il s'agit du cas général, et

seules quelques configurations particulières pour les bijections f et h ont été écartées. Comment évaluer la probabilité d'obtenir une collision lors de la requête suivante ?

Tout d'abord, les bijections f et h sont au centre du jeu : leur choix détermine toutes les structures des groupes, ainsi que les logarithmes discrets des éléments des groupes manipulés. Dans le cadre de la famille générique, ces bijections ont été choisies aléatoirement et uniformément dans l'ensemble de toutes les bijections de $(\mathbb{Z}/n\mathbb{Z})$ dans $B(n)$, mais certaines possibilités ont été écartées par les k premiers appels aux oracles. Les seuls couples de bijections envisageables (de manière équiprobable) sont les couples (f, h) compatibles avec les listes $(\mathsf{R}_k, \mathsf{S}_k)$ (voir définition 2.8).

L'existence ou l'absence de collisions après k ou $k+1$ appels aux oracles dépend en fait uniquement des images réciproques des r_k ou r_{k+1} valeurs x_i par f et des images réciproques des s_k ou s_{k+1} valeurs y_j par h. Dans le cas présent, ces images réciproques n'engendrent pas de collisions pendant les k premiers appels : elles sont donc simultanément cohérentes avec l'ensemble des fractions rationnelles de R_k et l'ensemble des fractions rationnelles de S_k (voir définition 2.9). L'existence d'une collision après l'appel suivant s'interprète inversement : ces images réciproques doivent ne plus être cohérentes avec l'ensemble des fractions rationnelles de R_{k+1}, ou celui des fractions rationnelles de S_{k+1}. L'évaluation de la probabilité de collision de l'algorithme revient donc à estimer la part d'ensembles d'images réciproques incohérentes à l'étape $k+1$ dans les ensembles d'images réciproques cohérentes à l'étape k.

2.4.2 Lemme fondamental

La preuve du lemme 2.1 suit exactement ce cheminement, en montrant dans un premier temps comment passer des bijections à leurs images réciproques, puis en calculant le ratio d'ensembles d'images réciproques qui basculent de la cohérence à l'incohérence à l'étape $k+1$.

Lemme 2.1 *On suppose que $n = p^\lambda$ où p est un nombre premier. Pour un entier k fixé, on considère un couple de listes $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision, et un $(k+1)^{\text{ème}}$ appel à l'un des oracles.*
On écrit toutes les fractions rationnelles de R_k et de S_k sous forme réduite, et on note d_1 le degré maximal des numérateurs de ces fractions rationnelles, et d_2 le degré maximal des dénominateurs de ces mêmes fractions rationnelles. Soit $d = d_1 + d_2$.
Lorsque $\rho_k + \sigma_k < \sqrt{2p/3d}$, la probabilité sur l'ensemble des couples de bijections compatibles avec $(\mathsf{R}_k, \mathsf{S}_k)$ que le nouvel appel à l'un des oracles conduise à des listes $(\mathsf{R}_{k+1}, \mathsf{S}_{k+1})$ présentant une collision est borné par :

$$\frac{6d\,(\rho_k + \sigma_k)}{2p - 3d\,(\rho_k + \sigma_k)^2}.$$

Preuve Soit $\mathcal{C}$ l'ensemble des $(r_{k+1} + s_{k+1})$-uplets qui sont simultanément cohérents avec l'ensemble des fractions rationnelles de R_k et celui des fractions rationnelles de S_k.

Pour tout couple de bijections (f, h) compatible avec les listes $(\mathsf{R}_k, \mathsf{S}_k)$, le $(r_{k+1} + s_{k+1})$-uplet $(f^{-1}(x_1), \ldots, f^{-1}(x_{r_{k+1}}), h^{-1}(y_1), \ldots, h^{-1}(y_{s_{k+1}}))$ est par définition dans $\mathcal{C}$, puisqu'il n'y a pas de collision dans $(\mathsf{R}_k, \mathsf{S}_k)$.

Réciproquement, pour tout $(\alpha_1, \ldots, \alpha_{r_{k+1}}, \beta_1, \ldots, \beta_{s_{k+1}})$ dans $\mathcal{C}$, on construit des couples de bijections (f, h) compatibles avec les listes $(\mathsf{R}_k, \mathsf{S}_k)$ et telles que $f^{-1}(x_i) = \alpha_i$ pour tout $i \in \{1, \ldots, r_{k+1}\}$ et $h^{-1}(y_j) = \beta_j$ pour tout $j \in \{1, \ldots, s_{k+1}\}$. En plus des r_{k+1} valeurs fixées pour f, la compatibilité avec les listes $(\mathsf{R}_k, \mathsf{S}_k)$ ajoute $\rho_k - r_k$ nouvelles valeurs fixées. De même, ce sont $\sigma_k - s_k$ nouvelles valeurs qui sont fixées pour h. Les autres valeurs sont totalement libres, ce qui donne $(n+r_k-r_{k+1}-\rho_k)!(n+s_k-s_{k+1}-\sigma_k)!$ couples de bijections compatibles avec les listes $(\mathsf{R}_k, \mathsf{S}_k)$ et vérifiant $f^{-1}(x_i) = \alpha_i$ pour $i \leqslant r_{k+1}$ et $h^{-1}(y_j) = \beta_j$ pour $j \leqslant s_{k+1}$.

Globalement, on obtient exactement $(\#\mathcal{C})(n + r_k - r_{k+1} - \rho_k)!(n + s_k - s_{k+1} - \sigma_k)!$ couples de bijections compatibles avec $(\mathsf{R}_k, \mathsf{S}_k)$.

Même si les ensembles R_{k+1} et S_{k+1} ne sont pas complètement définis (on ne connaît pas la réponse à la $(k+1)^{\text{ème}}$ requête), les fractions rationnelles de ces ensembles sont totalement déterminées. On définit alors l'ensemble $\mathcal{C}'$ des $(r_{k+1} + s_{k+1})$-uplets qui sont simultanément cohérents avec l'ensemble des fractions rationnelles de R_{k+1} et celui des fractions rationnelles de S_{k+1}. En suivant exactement le même raisonnement que précédemment, on dénombre exactement $(\#\mathcal{C}')(n+r_k-r_{k+1}-\rho_k)!(n+s_k-s_{k+1}-\sigma_k)!$ couples de bijections compatibles avec les listes $(\mathsf{R}_k, \mathsf{S}_k)$ et conduisant à un couple de listes $(\mathsf{R}_{k+1}, \mathsf{S}_{k+1})$ sans collision.

La probabilité que le $(k+1)^{\text{ème}}$ appel à l'oracle conduise à des listes $(\mathsf{R}_{k+1}, \mathsf{S}_{k+1})$ avec collision, à partir de listes $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision, est donc exactement $(\#\mathcal{C} - \#\mathcal{C}')/\#\mathcal{C}$. D'après la remarque 2.2, le $(k+1)^{\text{ème}}$ appel à l'oracle correspond à au plus trois nouvelles fractions rationnelles dans R_{k+1} ou S_{k+1}, par rapport aux listes R_k et S_k. Mais parmi elles, seule une, notée P, peut engendrer un collision (celles correspondant aux nouvelles variables ne peuvent par construction pas collisionner).

Si une telle fraction rationnelle P est ajoutée dans R_{k+1}, alors un $(r_{k+1} + s_{k+1})$-uplet $(\alpha_1, \ldots, \alpha_{r_{k+1}}, \beta_1, \ldots, \beta_{s_{k+1}})$ est incohérent avec les fractions rationnelles de R_{k+1} si et seulement s'il existe une fraction rationnelle P_R dans la liste R_k telle que $(P - P_R)(\alpha_1, \ldots, \alpha_{r_{k+1}}, \beta_1 \ldots, \beta_{s_{k+1}}) = 0$. Il y a par définition ρ_k fractions rationnelles dans R_k, et pour chacune d'elles, la différence $P - P_R$ écrite sous forme réduite a un numérateur qui a au plus $(d.n^{r_{k+1}+s_{k+1}}/p)$ racines, d'après [Sch80] lemme 1.

De même, si la nouvelle fraction rationnelle P est ajoutée dans S_{k+1}, il y a exactement σ_k fractions rationnelles dans S_k, chacune d'elles pouvant collisionner avec P sur au plus $(d.n^{r_{k+1}+s_{k+1}}/p)$ valeurs.

On obtient ainsi l'inégalité suivante : $\#\mathcal{C} - \#\mathcal{C}' \leqslant d\,(\rho_k + \sigma_k)\, n^{r_{k+1}+s_{k+1}}\,/\,p$.

En utilisant à nouveau [Sch80] lemme 1, mais cette fois-ci sur les $(\rho_k(\rho_k - 1) + \sigma_k(\sigma_k - 1))/2$ numérateurs des différences de fractions rationnelles de R_k ou de S_k, on obtient une borne inférieure pour la taille de $\mathcal{C}$:

$$\#\mathcal{C} \geqslant \frac{n!}{(n - r_{k+1} - s_{k+1})!} \;-\; (\rho_k(\rho_k - 1) + \sigma_k(\sigma_k - 1))\,.\,\frac{d\,.\,n^{r_{k+1}+s_{k+1}}}{2p}.$$

Par hypothèse, on a $r_{k+1} + s_{k+1} \leqslant \sqrt{n}$. Un calcul rapide permet d'en déduire :

$$\frac{n!}{(n - r_{k+1} - s_{k+1})!} \geqslant \frac{n^{r_{k+1}+s_{k+1}}}{3}.$$

On obtient alors l'inégalité suivante : $\#\mathcal{C} \geqslant n^{r_{k+1}+s_{k+1}} . \big(1/3 - d\,(\rho_k + \sigma_k)^2/(2p)\big)$.

La borne supérieure de $(\#\mathcal{C} - \#\mathcal{C}')$ et la borne inférieure de $\#\mathcal{C}$ nous conduisent naturellement à une borne supérieure de la probabilité de collision dans les listes $(\mathsf{R}_{k+1}, \mathsf{S}_{k+1})$, à partir de listes $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision :

$$\frac{\#\mathcal{C} - \#\mathcal{C}'}{\#\mathcal{C}} \leqslant \frac{6d\,(\rho_k + \sigma_k)}{2p - 3d\,(\rho_k + \sigma_k)^2}.$$

□

Le lemme 2.1 est fondamental pour l'étude de problèmes dans la famille générique de groupes cycliques avec couplage. En effet, il suffit d'interpréter les problèmes étudiés sous la forme de conditions de collisions pour utiliser directement le résultat de ce lemme et obtenir ainsi une probabilité de réussite d'un algorithme.

2.4.3 Illustration sur le problème bilinéaire Diffie-Hellman

On illustre l'utilisation du lemme 2.1 sur le problème bilinéaire Diffie-Hellman. Dans les deux corollaires suivants, on suppose que l'ordre des groupes considérés est une puissance d'un nombre premier : $n = p^\lambda$. Ceci permet d'appliquer le lemme immédiatement. Le cas général d'un ordre composite est traité dans le théorème 2.1, qui assemble les résultats des corollaires et les généralise.

Soit $\mathcal{A}$ un algorithme résolvant le problème bilinéaire Diffie-Hellman. Notre modèle de calcul suppose que l'algorithme $\mathcal{A}$ est une machine de Turing probabiliste, dont la complexité en temps est définie par le nombre d'appels aux oracles qu'il réalise. il reçoit en entrées la taille n des groupes, les éléments neutres, 0_f et 0_h, et les générateurs, g_f et g_h, des deux groupes, ainsi que trois éléments du groupe $B(n)_f$: x_1, x_2 et x_3.

Au début du jeu, les listes R_0 et S_0 sont donc définies par :
- $\mathsf{R}_0 = \{(0_f, 0), (g_f, 1), (x_1, X_1), (x_2, X_2), (x_3, X_3)\}$;
- $\mathsf{S}_0 = \{(0_h, 0), (g_h, 1)\}$.

Corollaire 2.1 *Pour $n = p^\lambda$, soit $k \leqslant (\sqrt{p/3} - 5)/3$. On considère des suites de listes R et S basées sur des polynômes de degré au plus 2. La probabilité qu'il y ait une collision après k appels aux oracles, c'est-à-dire dans les listes $(\mathsf{R}_k, \mathsf{S}_k)$, est majorée par :*

$$\frac{2\,(3k+4)^2}{p - 3\,(3k+4)^2}.$$

Preuve D'après la remarque 2.2, au plus trois ajouts ont lieu dans les listes à chaque appel à un oracle. On en déduit que pour tout $i \in \{1, \ldots, k\}$, $\rho_i + \sigma_i \leqslant 3i + 7$. Ainsi, $6\,(\rho_i + \sigma_i)/(p - 3\,(\rho_i + \sigma_i)^2)$ est majoré par $6\,(3i + 7)/(p - 3\,(3i + 7)^2)$.

La probabilité que les listes $(\mathsf{R}_k, \mathsf{S}_k)$ soient sans collision après k appels aux oracles est égale au produit des probabilités d'absence de collision dans les listes $(\mathsf{R}_{i+1}, \mathsf{S}_{i+1})$, sachant que les listes $(\mathsf{R}_i, \mathsf{S}_i)$ sont sans collision, pour i allant de 0 à $k - 1$. Ces probabilités « locales » sont exactement celles calculées par le lemme 2.1.

La borne sur k se traduit par l'encadrement suivant : $0 < 6\,(3k+4) < p-3\,(3k+4)^2$, qui assure que la borne inférieure donnée par le lemme 2.1 pour l'indice $i = k - 1$ reste positive.

On peut alors réaliser la minoration suivante de la probabilité de non-collision dans les listes $(\mathsf{R}_k, \mathsf{S}_k)$:

$$\prod_{i=0}^{k-1} \Big(1 - \frac{6\,(3i + 7)}{p - 3\,(3i + 7)^2}\Big) \geqslant \Big(1 - \frac{6\,(3k + 4)}{p - 3\,(3k + 4)^2}\Big)^k \geqslant 1 - \frac{2\,(3k + 4)^2}{p - 3\,(3k + 4)^2}.$$

Une borne supérieure de la probabilité de collision dans les listes $(\mathsf{R}_k, \mathsf{S}_k)$ s'en déduit immédiatement : $2\,(3k + 4)^2/(p - 3\,(3k + 4)^2)$. □

Le corollaire 2.1 donne une probabilité de non-collision dans les listes $(\mathsf{R}_k, \mathsf{S}_k)$. Le corollaire suivant évalue la probabilité de succès d'un algorithme sur le problème bilinéaire Diffie-Hellman dans ce cas de non-collision. Cette évaluation est en fait basée une nouvelle fois sur un calcul de collision :

- l'algorithme peut retourner un élément qui n'est jamais apparu dans les entrées et les réponses des oracles $+_h$, $-_h$ ou $e_{f,h}$ (donc pas dans la liste S_k), mais tous les éléments vérifiant cette propriété sont indistinguables ;
- l'algorithme peut à l'inverse retourner une valeur apparaissant dans la liste S_k, mais cela revient à parier sur l'existence d'une collision suite à une requête correspondant au problème Diffie-Hellman (de polynôme égal à $X_1X_2X_3$).

Corollaire 2.2 *On considère un algorithme $\mathcal{A}$ sur le problème bilinéaire Diffie-Hellman dans la famille générique de groupes cycliques avec couplage. Pour un ordre des groupes puissance d'un nombre premier, $n = p^\lambda$, lorsque $k < (\sqrt{2p/9} - 7)/3$, la probabilité de succès de $\mathcal{A}$, après k appels aux oracles et avec des listes $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision, est majorée par :*

$$\frac{18\,(3k + 7)}{2p - 9\,(3k + 7)^2}.$$

Preuve On note z la réponse donnée par l'algorithme $\mathcal{A}$ au problème Diffie-Hellman, après k appels aux oracles ayant conduit à des suites $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision. On rappelle que par construction (voir remarque 2.2), après k appels aux oracles, on a les bornes : $\sigma_k \leqslant 3k + 2$ et $\rho_k + \sigma_k \leqslant 3k + 7$.

Si z n'apparaît dans aucun couple de S_k, alors z est indistinguable de tout autre élément de $B(n)$ avec cette même propriété. Ainsi, dans ce cas, l'algorithme $\mathcal{A}$ résout le problème avec une probabilité au plus égale à $1/(p-\sigma_k) \leqslant 1/(p-3k-2)$.

Si z apparaît dans un couple de S_k, lorsque cette réponse est valide, l'ajout du couple $(z, X_1X_2X_3)$ à la liste S_k conduit à une collision. Ainsi, une requête « virtuelle » (car non permise dans le modèle) correspondant au polynôme $X_1X_2X_3$ conduit à une collision, depuis des listes $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision par hypothèse. Le lemme 2.1 s'applique donc immédiatement, et on obtient la borne supérieure suivante :

$$\frac{18\,(\rho_k+\sigma_k)}{2p-9\,(\rho_k+\sigma_k)^2} \leqslant \frac{18\,(3k+7)}{2p-9\,(3k+7)^2}.$$

La probabilité de réussite de $\mathcal{A}$ est majorée par le maximum de ces deux bornes, qui est toujours égale à la seconde borne. □

On recombine ces deux corollaires sous la forme du théorème 2.1. On en profite pour généraliser le résultat au cas d'un ordre quelconque pour les groupes.

Théorème 2.1 *On considère un algorithme $\mathcal{A}$ sur le problème bilinéaire Diffie-Hellman dans la famille générique de groupes cycliques avec couplage, disposant d'une puissance calculatoire infinie, et capable d'appeler les oracles de manière probabiliste. Sur des groupes d'ordre divisible par un nombre premier p, lorsque le nombre k d'appels aux oracles vérifie $k < (\sqrt{2p/9}-7)/3$, la probabilité de succès de $\mathcal{A}$ est majorée par :*

$$\frac{2\,(3k+4)^2}{p-3\,(3k+4)^2} + \frac{18\,(3k+7)}{2p-9\,(3k+7)^2}.$$

Preuve Soit n l'ordre commun des groupes. On suppose dans un premier temps que $n = p^\lambda$, avec p un nombre premier :

- la probabilité que les listes $(\mathsf{R}_k, \mathsf{S}_k)$ présentent une collision est bornée par le corollaire 2.1 ;
- lorsque les listes $(\mathsf{R}_k, \mathsf{S}_k)$ sont sans collision, la probabilité de succès de l'algorithme $\mathcal{A}$ est bornée par le corollaire 2.2.

La somme des bornes données par ces deux corollaires est donc une borne de la probabilité globale de succès de l'algorithme $\mathcal{A}$.

La généralisation au cas d'un ordre composite $n = p^\lambda q$, avec p premier et q non-divisble par p, se fait par réduction. Ainsi, à partir d'un algorithme $\mathcal{A}_n$ basé sur des groupes d'ordre n, on construit un algorithme $\mathcal{A}_{p^\lambda}$ pour des groupes d'ordre p^λ ayant une probabilité de succès au moins égale :

1. $\mathcal{A}_{p^\lambda}$ choisit aléatoirement deux bijection ϕ_1, ϕ_2 de $B(p^\lambda) \times \mathbb{Z}/q\mathbb{Z}$ dans $B(n)$;
2. $\mathcal{A}_{p^\lambda}$ choisit aléatoirement trois éléments de $\mathbb{Z}/q\mathbb{Z}$: α_1, α_2, et α_3 ;
3. $\mathcal{A}_{p^\lambda}$ reçoit les entrées : $(0_f, g_f, 0_h, g_h) \in B(p^\lambda)^4$ et $(x_1, x_2, x_3) \in B(p^\lambda)^3$;
4. $\mathcal{A}_{p^\lambda}$ calcule les entrées pour $\mathcal{A}_n$: $0'_f = \phi_1(0_f, 0)$, $0'_h = \phi_2(0_h, 0)$, $g'_f = \phi_1(g_f, 1)$, $g'_h = \phi_2(g_h, 1)$, $x'_1 = \phi_1(x_1, \alpha_1)$, $x'_2 = \phi_1(x_2, \alpha_2)$ et $x'_3 = \phi_1(x_3, \alpha_3)$;

5. $\mathcal{A}_n$ calcule une réponse z' au problème bilinéaire Diffie-Hellman à partir des entrées calculées précédemment : $(0'_f, g'_f, 0'_h, g'_h)$ et (x'_1, x'_2, x'_3) ;
6. à partir de z, $\mathcal{A}_{p^\lambda}$ calcule $(z, \alpha) = {\phi_2}^{-1}(z')$, et donne z comme réponse.

Pour l'étape 1, on remarque qu'il existe une bijection naturelle ϕ de $\mathbb{Z}/p^\lambda\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$ dans $\mathbb{Z}/n\mathbb{Z}$ donnée par $\phi(a, b) = a + p^\lambda b$. Ainsi, le choix des bijections ϕ_1 et ϕ_2 revient à choisir aléatoirement deux permutations sur $B(n)$. En fait, au lieu de générer complètement ces deux permutations, on se contente de choisir aléatoirement de nouvelles valeurs pour des requêtes fraîches à ces permutations.

L'algorithme $\mathcal{A}_{p^\lambda}$ doit également construire les oracles utilisés par $\mathcal{A}_n$ à partir des oracles auxquels il a accès :

$+'_f : (\phi_1(a_1, b_1), \phi_1(a_2, b_2)) \in B(n)^2 \mapsto \phi_1(a_1 +_f a_2, b_1 + b_2) \in B(n)$;
$-'_f : \phi_1(a, b) \in B(n) \mapsto \phi_1(-_f a, -b) \in B(n)$;
$+'_h : (\phi_2(a_1, b_1), \phi_2(a_2, b_2)) \in B(n)^2 \mapsto \phi_2(a_1 +_h a_2, b_1 + b_2) \in B(n)$;
$-'_h : \phi_2(a, b) \in B(n) \mapsto \phi_2(-_h a, -b) \in B(n)$;
$e'_{f,g} : (\phi_1(a_1, b_1), \phi_1(a_2, b_2)) \in B(n)^2 \mapsto \phi_2(e_{f,g}(a_1, a_2), b_1.b_2) \in B(n)$.

On vérifie facilement que l'ensemble $B(n)$ a une structure de groupe avec les lois $+'_f$ et $-'_f$, dont $0'_f$ est l'élément neutre et g'_f est un générateur. De même, l'ensemble $B(n)$ a une seconde structure de groupe avec les lois $+'_h$ et $-'_h$, dont $0'_h$ est l'élément neutre et g'_h est un générateur. $e'_{f,h}$ est un couplage parfait entre ces structures de groupes vérifiant : $e'_{f,h}(g'_f, g'_f) = g'_h$.

Si $z' = \phi_2(z, \alpha)$ est la solution au problème bilinéaire Diffie-Hellman soumis à l'algorithme $\mathcal{A}_n$, alors, par construction, z est la solution au problème bilinéaire Diffie-Hellman soumis à l'algorithme $\mathcal{A}_{p^\lambda}$. La probabilité de succès de $\mathcal{A}_{p^\lambda}$ est donc au moins égale à celle de $\mathcal{A}_n$, ce qui clôt la preuve par réduction. □

Une conséquence immédiate de ce théorème est que si le nombre d'appels aux oracles est petit devant un facteur premier p de l'ordre des groupes, la probabilité de succès d'un algorithme sur le problème bilinéaire Diffie-Hellman est au plus de l'ordre de k^2/p dans la famille générique de groupes cycliques avec couplage.

Grâce à la réduction polynomiale du problème bilinéaire Diffie-Hellman sur le problème du logarithme discret, le théorème 2.1 peut directement se transposer au problème du logarithme discret. Ainsi, l'ajout du couplage n'aide pas significativement à la résolution du problème du logarithme discret. De même, dans la famille générique de groupes cycliques avec couplage, le problème Diffie-Hellman est difficile dans le groupe $B(n)_f$, tout comme dans le groupe $B(n)_h$.

2.4.4 Illustration sur le problème q-BDHI

On souhaite montrer l'aspect général de la technique de preuve présentée en section 2.4.3 sous la forme de deux corollaires du lemme 2.1, regroupés ensuite dans un théorème. On présente donc ici l'étude du problème de la q-inversion bilinéaire Diffie-Hellman, introduit dans [BB04a].

Définition 2.10 (Problème q-BDHI) *Soit $(\Omega, (L_\gamma)_{\gamma\in\Omega_1}, (M_\delta)_{\delta\in\Omega_2}, (e_\alpha)_{\alpha\in\Omega})$ une famille de représentations de groupes cycliques avec couplage sur deux langages L et M, basée exclusivement sur des groupes d'ordre premier. Dans cette famille, un algorithme résolvant la q-inversion bilinéaire Diffie-Hellman (problème q-BDHI) calcule l'élément $(1/\nu)\, g_\delta$ dans le groupe M_δ à partir des données $(\gamma, \delta) \in \Omega$, $(\nu\, g_\gamma, \nu^2\, g_\gamma, \ldots, \nu^q\, g_\gamma) \in (L_\gamma)^q$.*

On considère un algorithme $\mathcal{A}$ résolvant le problème q-BDHI dans la famille générique de groupes cycliques avec couplage, restreinte aux groupes d'ordre premier. Il reçoit en entrées un nombre premier p, un quadruplet $(0_f, g_f, 0_h, g_h) \in B(p)^4$ et un q-uplet $(\nu\, g_f, \ldots, \nu^q\, g_f) \in B(p)^q$ où $\nu \in (\mathbb{Z}/p\mathbb{Z})^*$. Ainsi, les valeurs initiales des suites de listes R et S sont $\mathsf{R}_0 = \{(0_f, 0), (g_f, 1), (\nu\, g_f, X_1), (\nu^2\, g_f, {X_1}^2), \ldots, (\nu^q\, g_f, {X_1}^q)\}$ et $\mathsf{S}_0 = \{(0_h, 0), (g_h, 1)\}$. Les listes sont ensuite construites au fur et à mesure des appels aux oracles, comme présenté en section 2.4.1.

Corollaire 2.3 *Soit p l'ordre premier des groupes, soit $k \leqslant (\sqrt{p/(3q)} - q - 5)/3$. On considère des suites de listes R et S basées sur des polynômes de degré au plus $2q$. La probabilité qu'il y ait une collision après k appels aux oracles, c'est-à-dire dans les listes $(\mathsf{R}_k, \mathsf{S}_k)$, est majorée par :*

$$\frac{2q\,(3k+q+1)^2}{p - 3q\,(3k+q+1)^2}.$$

Preuve La preuve est semblable à celle du corollaire 2.1. En effet, en utilisant le lemme 2.1, on obtient la minoration suivante pour la probabilité de non-collision :

$$\begin{aligned}\prod_{i=0}^{k-1}\Big(1 - \frac{6q\,(3i+q+4)}{p - 3q(3i+q+4)^2}\Big) &\geqslant \Big(1 - \frac{6q(3k+q+1)}{p - 3q(3k+q+1)^2}\Big)^k \\ &\geqslant 1 - \frac{2q\,(3k+q+1)^2}{p - 3q\,(3k+q+1)^2}.\end{aligned}$$

□

Corollaire 2.4 *On considère un algorithme $\mathcal{A}$ sur le problème q-BDHI dans la famille générique de groupes cycliques avec couplage. Pour un ordre des groupes premier p, lorsque $k < (\sqrt{2p/(6q+3)} - q - 4)/3$, la probabilité de succès de $\mathcal{A}$, après k appels aux oracles et avec des listes $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision, est majorée par :*

$$\frac{6\,(2q+1)(3k+q+4)}{2p - 3\,(2q+1)(3k+q+4)^2}.$$

Preuve La preuve est semblable à celle du corollaire 2.2. On note z la réponse donnée par l'algorithme $\mathcal{A}$ au problème q-BDHI, après k appels aux oracles ayant conduit à des suites $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision.

Si z n'apparaît dans aucun couple de S_k, alors z est indistinguable de tout autre élément de $B(p)$ avec cette même propriété. Ainsi, dans ce cas, l'algorithme $\mathcal{A}$ résout le problème avec une probabilité au plus égale à $1/(p-\sigma_k) \leqslant 1/(p-3k-2)$.

Si z apparaît dans un couple de S_k, lorsque cette réponse est valide, une requête « virtuelle » correspondant à la fraction rationnelle $1/X_1$ conduit à une collision, depuis des listes $(\mathsf{R}_k, \mathsf{S}_k)$ sans collision. Le lemme 2.1 donne la borne supérieure suivante pour la probabilité de cet événement :

$$\frac{6\,(2q+1)(\rho_k+\sigma_k)}{2p-3\,(2q+1)(\rho_k+\sigma_k)^2} \leqslant \frac{6\,(2q+1)(3k+q+4)}{2p-3\,(2q+1)(3k+q+4)^2}.$$

□

Les corollaires 2.3 et 2.4, se combinent pour donner le théorème suivant :

Théorème 2.2 *On considère un algorithme $\mathcal{A}$ sur le problème q-BDHI dans la famille générique de groupes cycliques avec couplage, disposant d'une puissance calculatoire infinie, et capable d'appeler les oracles de manière probabiliste. Sur des groupes d'ordre premier p, lorsque le nombre k d'appels aux oracles vérifie $k < (\sqrt{2p/(6q+3)}-q-4)/3$, la probabilité de succès de $\mathcal{A}$ est majorée par :*

$$\frac{2q\,(3k+q+1)^2}{p-3q\,(3k+q+1)^2} + \frac{6\,(2q+1)(3k+q+4)}{2p-3\,(2q+1)(3k+q+4)^2}.$$

Ainsi, lorsque le nombre d'appels aux oracles est petit devant l'ordre des groupes, la probabilité de succès d'un algorithme sur le problème q-BDHI dans la famille générique de groupes cycliques avec couplage est au plus de l'ordre de k^2/p, avec une dépendance linéaire en q.

2.5 Familles pseudo-aléatoires de groupes

Dans cette partie, on introduit la notion de famille pseudo-aléatoire de groupes cycliques. Schématiquement, une famille pseudo-aléatoire de groupes cycliques est similaire à la famille générique de groupes cycliques, à la différence que la loi de groupe n'est pas choisie aléatoirement parmi l'ensemble de toutes les lois possibles. La loi de groupe suit une distribution particulière, qui est calculatoirement indistinguable de la distribution uniforme. On construit de telles familles à partir de familles pseudo-aléatoires de permutations.

2.5.1 Famille pseudo-aléatoire de permutations

Soit n un entier non-nul et $B(n)$ l'ensemble des écritures binaires des entiers entre 0 et $n-1$. On considère un ensemble $\mathfrak{P}$ de permutations sur $B(n)$. On note $f \leftarrow \mathfrak{P}$ le tirage aléatoire uniforme de f dans $\mathfrak{P}$. Un distingueur $\mathcal{D}$ est une machine de Turing qui accède à une permutation sur $B(n)$ via un oracle, et qui répond par un bit.

Dans le contexte de l'indistinguabilité forte (*strong indistinguishability*) entre deux familles de permutations, $\mathfrak{P}_1$ et $\mathfrak{P}_2$, un distingueur $\mathcal{D}$ a accès aux oracles d'une permutation f et de son inverse f^{-1}, où $f \leftarrow \mathfrak{P}_1$ ou $f \leftarrow \mathfrak{P}_2$ (voir [LR88] pour plus de détails). Lorsque $\mathcal{D}$ dispose au plus d'un temps t et de q appels aux oracles, son avantage est défini par la formule suivante :

$$\mathrm{Adv}^{\text{s-ind}}_{\mathfrak{P}_1,\mathfrak{P}_2}(\mathcal{D},t,q) = \left| \Pr_{f\leftarrow\mathfrak{P}_1}[{\mathcal{D}_{t,q}}^{f,f^{-1}} = 1] - \Pr_{f\leftarrow\mathfrak{P}_2}[{\mathcal{D}_{t,q}}^{f,f^{-1}} = 1] \right|.$$

Les familles de permutations $\mathfrak{P}_1$ et $\mathfrak{P}_2$ sont dites (ϵ,t,q)-fortement indistinguables si pour tout distingueur $\mathcal{D}$, l'avantage $\mathrm{Adv}^{\text{s-ind}}_{\mathfrak{P}_1,\mathfrak{P}_2}(\mathcal{D},t,q)$ est inférieur à ϵ.

Une famille de permutations $\mathfrak{P}$ est dite (ϵ,t,q)-fortement pseudo-aléatoire si elle est (ϵ,t,q)-fortement indistinguable de l'ensemble $\mathfrak{S}_n$ de toutes les permutations sur $B(n)$.

2.5.2 Famille pseudo-aléatoire de groupes cycliques

Définition 2.11 (Famille pseudo-aléatoire de groupes cycliques) *Soit $\mathfrak{P}$ une famille de permutations (ϵ,t,q)-fortement pseudo-aléatoire sur $B(n)$. La famille pseudo-aléatoire de groupes cycliques associée à $\mathfrak{P}$ est l'ensemble des structures de groupe sur $B(n)$ définies par les permutations f de $\mathfrak{P}$: élément neutre défini par 0_f, générateur défini par $f(1)$ et lois de groupes $+_f$ et $-_f$ construites comme indiqué en section 2.2.3.*

Comme pour la famille générique de groupes cycliques, les lois de groupe ne sont données que via des oracles De cette manière, il est clair que la famille générique de groupes cycliques est une famille pseudo-aléatoire de groupes cycliques. Ainsi, la notion de famille pseudo-aléatoire de groupes est une généralisation de la notion de famille générique.

On peut maintenant définir l'avantage d'un algorithme sur un problème basé sur le logarithme discret dans une famille pseudo-aléatoire de groupes cycliques. Soit $\mathcal{A}$ un algorithme sur la famille pseudo-aléatoire de groupes cyclique associée à la famille de permutations $\mathfrak{P}$. Ses avantages sur le problème du logarithme discret, le problème Diffie-Hellman et le problème décisionnel Diffie-Hellman, pour un temps de calcul t et q appels aux oracles de groupe, sont définis par :

$$\mathrm{Adv}^{\mathrm{DL}}_{\mathfrak{P}}(\mathcal{A},t,q) = \Pr_{f\leftarrow\mathfrak{P},\,x\in B(n)}[{\mathcal{A}_{t,q}}^{+_f,-_f}(f(0),f(1),x) = \log_{f(1)}(x)],$$

$$\mathrm{Adv}^{\mathrm{DH}}_{\mathfrak{P}}(\mathcal{A},t,q) = \Pr_{f\leftarrow\mathfrak{P},\,(x,y)\in B(n)^2}[{\mathcal{A}_{t,q}}^{+_f,-_f}(f(0),f(1),x,y) = \log_{f(1)}(x)\,.\,y],$$

$$\mathrm{Adv}^{\mathrm{DDH}}_{\mathfrak{P}}(\mathcal{A},t,q) = \Big| \Pr_{f\leftarrow\mathfrak{P},\,(x,y)\in B(n)^2}[{\mathcal{A}_{t,q}}^{+_f,-_f}(f(0),f(1),x,y,\log_{f(1)}(x)\,.\,y) = 1] - \Pr_{f\leftarrow\mathfrak{P},\,(x,y,z)\in B(n)^3}[{\mathcal{A}_{t,q}}^{+_f,-_f}(f(0),f(1),x,y,z) = 1]\Big|.$$

Les maxima de ces avantages sur l'ensemble des algorithmes s'exécutant en temps t et avec au plus q appels aux oracles de groupes sont respectivement notés $\mathrm{Adv}^{\mathrm{DL}}_{\mathfrak{P}}(t,q)$, $\mathrm{Adv}^{\mathrm{DH}}_{\mathfrak{P}}(t,q)$ et $\mathrm{Adv}^{\mathrm{DDH}}_{\mathfrak{P}}(t,q)$. Les théorèmes 2.3 et 2.4 donnent des bornes supérieures à ces avantages, lorsque $\mathfrak{P}$ est une famille de permutations (ϵ,t,q)-fortement pseudo-aléatoire sur $B(n)$.

2.5.3 Problèmes Diffie-Hellman et logarithme discret

Le théorème suivant traite les cas de problèmes calculatoires, c'est-à-dire le problème du logarithme discret et le problème Diffie-Hellman.

Théorème 2.3 *Soit $\mathfrak{P}$ une famille de permutations (ϵ, t, q)-fortement pseudo-aléatoire sur $B(n)$. On a les deux inégalités suivantes :*

$$Adv^{DL}_{\mathfrak{P}}(t, q/3-1) \leqslant Adv^{DL}_{\mathfrak{S}_n}(t, q/3-1) + \epsilon,$$
$$Adv^{DH}_{\mathfrak{P}}(t, (q-4)/3) \leqslant Adv^{DH}_{\mathfrak{S}_n}(t, (q-4)/3) + \epsilon.$$

Preuve On considère un algorithme $\mathcal{A}$ pour le problème du logarithme discret sur la famille pseudo-aléatoire de groupes cycliques basée sur la famille de permutations $\mathfrak{P}$. À partir de cet algorithme $\mathcal{A}$, on construit un distingueur $\mathcal{D}$ sur les familles de permutations $\mathfrak{P}$ et $\mathfrak{S}_n$. On obtient alors une réduction.

Le distingueur $\mathcal{D}$ reçoit l'accès à deux oracles, implémentant les permutations f et f^{-1}, où f a été choisi aléatoirement dans $\mathfrak{P}$ ou dans $\mathfrak{S}_n$. Le distingueur $\mathcal{D}$ construit une instance du problème du logarithme discret en choisissant aléatoirement un élément $r \in \mathbb{Z}/n\mathbb{Z}$: l'élément neutre du groupe est $f(0)$, le générateur considéré est $f(1)$ et l'élément dont le logarithme doit être calculé est $f(r)$.

Pour simuler l'appel aux oracles des lois de groupe, le distingueur $\mathcal{D}$ utilise les oracles donnant accès aux permutations f et f^{-1} : $x +_f y = f\left(f^{-1}(x) + f^{-1}(y)\right)$, $-_f x = f\left(-f^{-1}(x)\right)$.

À la fin du jeu, l'algorithme $\mathcal{A}$ donne une réponse au problème du logarithme discret. Si cette réponse est correcte (c'est-à-dire égale à r), le distingueur $\mathcal{D}$ retourne 1, dans le cas contraire il retourne 0.

Si la permutation f a été choisie dans $\mathfrak{P}$, la probabilité que l'algorithme $\mathcal{A}$ réponde correctement est exactement son avantage sur le problème du logarithme discret dans la famille pseudo-aléatoire de groupes cycliques basée sur $\mathfrak{P}$.

Dans le cas contraire, lorsque la permutation f a été choisie dans $\mathfrak{S}_n$, la probabilité que l'algorithme $\mathcal{A}$ réponde correctement est exactement son avantage sur le problème du logarithme discret dans la famille générique de groupes cycliques d'ordre n.

L'avantage du distingueur $\mathcal{D}$ est donc exactement la différence d'avantage de $\mathcal{A}$ entre ces deux cas. Ainsi, si $\mathcal{A}$ tourne en temps t et réalise au plus $(q/3-1)$ appels aux oracles de groupe, cet avantage est par définition de $\mathfrak{P}$ majoré par ϵ. On en déduit que l'avantage de tout algorithme $\mathcal{A}$ sur le problème du logarithme discret dans la famille pseudo-aléatoire de groupes cyclique basée sur $\mathfrak{P}$ est majoré par son avantage sur ce même problème dans la famille générique de groupes cycliques d'ordre n, auquel on ajoute ϵ.

Cette preuve se transpose immédiatement dans le cadre du problème Diffie-Hellman, avec seulement un appel initial supplémentaire à l'oracle de f, dû aux entrées plus nombreuses du problème. □

2.5.4 Problème décisionnel Diffie-Hellman

Dans l'analogue du théorème 2.3 pour le problème décisionnel Diffie-Hellman, la contribution ϵ du caractère pseudo-aléatoire de la famille de permutations est multipliée par 2 dans la borne de l'avantage de l'algorithme :

Théorème 2.4 *Soit $\mathfrak{P}$ une famille de permutations (ϵ, t, q)-fortement pseudo-aléatoire sur $B(n)$. On a l'inégalité suivante :*

$$Adv_{\mathfrak{P}}^{DDH}(t,(q-5)/3) \leqslant Adv_{\mathfrak{S}_n}^{DDH}(t,(q-5)/3) + 2\epsilon.$$

Preuve On considère un algorithme $\mathcal{A}$ pour le problème décisionnel Diffie-Hellman sur la famille pseudo-aléatoire de groupes cycliques basée sur la famille de permutations $\mathfrak{P}$. À partir de cet algorithme $\mathcal{A}$, on construit un distingueur $\mathcal{D}$ sur les familles de permutations $\mathfrak{P}$ et $\mathfrak{S}_n$. On obtient alors une réduction. Soit b le bit à deviner dans le problème décisionnel Diffie-Hellman :

$$\mathrm{Adv}_{\mathfrak{P}}^{\mathrm{DDH}}(\mathcal{A},t,q) = \left| \Pr_{f\leftarrow\mathfrak{P}}[{\mathcal{A}_{t,q}}^{+_f,-_f} = 1 \,/\, b=1] - \Pr_{f\leftarrow\mathfrak{P}}[{\mathcal{A}_{t,q}}^{+_f,-_f} = 1 \,/\, b=0] \right|.$$

$$\mathrm{Adv}_{\mathfrak{S}_n}^{\mathrm{DDH}}(\mathcal{A},t,q) = \left| \Pr_{f\leftarrow\mathfrak{S}_n}[{\mathcal{A}_{t,q}}^{+_f,-_f} = 1 \,/\, b=1] - \Pr_{f\leftarrow\mathfrak{S}_n}[{\mathcal{A}_{t,q}}^{+_f,-_f} = 1 \,/\, b=0] \right|.$$

Le distingueur $\mathcal{D}$ reçoit l'accès à deux oracles, implémentant les permutations f et f^{-1}, où f a été choisi aléatoirement dans $\mathfrak{P}$ ou dans $\mathfrak{S}_n$. Le distingueur $\mathcal{D}$ construit une instance du problème décisionnel Diffie-Hellman en choisissant aléatoirement un bit b et trois éléments x, y et z dans $\mathbb{Z}/n\mathbb{Z}$. L'élément neutre du groupe est $f(0)$, le générateur considéré est $f(1)$. Si $b=1$, l'instance contient de plus le triplet $(f(x), f(y), f(x.y))$. Si $b=0$, l'instance contient en remplacement le triplet $(f(x), f(y), f(z))$.

Pour simuler l'appel aux oracles des lois de groupe, le distingueur $\mathcal{D}$ utilise les oracles donnant accès aux permutations f et f^{-1} : $x +_f y = f\left(f^{-1}(x) + f^{-1}(y)\right)$, $-_f x = f\left(-f^{-1}(x)\right)$.

À la fin du jeu, l'algorithme $\mathcal{A}$ donne une réponse au problème décisionnel Diffie-Hellman. Si cette réponse est correcte (c'est-à-dire égale à b), le distingueur $\mathcal{D}$ retourne 1, dans le cas contraire il retourne 0.

On a par définition :

$$\mathrm{Adv}_{\mathfrak{P},\mathfrak{S}_n}^{\text{s-ind}}(\mathcal{D},t,q) = \left| \Pr_{f\leftarrow\mathfrak{P}}[{\mathcal{D}_{t,q}}^{f,f^{-1}} = 1] - \Pr_{f\leftarrow\mathfrak{S}_n}[{\mathcal{D}_{t,q}}^{f,f^{-1}} = 1] \right|.$$

Par construction du distingueur $\mathcal{D}$, cela signifie :

$$\begin{aligned}\mathrm{Adv}_{\mathfrak{P},\mathfrak{S}_n}^{\text{s-ind}}(\mathcal{D},t,q) = \Bigg| &\Pr_{f\leftarrow\mathfrak{P}}[{\mathcal{A}_{t,(q-5)/3}}^{+_f,-_f} = 1 \,/\, b=1] + \Pr_{f\leftarrow\mathfrak{P}}[{\mathcal{A}_{t,(q-5)/3}}^{+_f,-_f} = 0 \,/\, b=0] \\ &- \Pr_{f\leftarrow\mathfrak{S}_n}[{\mathcal{A}_{t,(q-5)/3}}^{+_f,-_f} = 1 \,/\, b=1] - \Pr_{f\leftarrow\mathfrak{S}_n}[{\mathcal{A}_{t,(q-5)/3}}^{+_f,-_f} = 0 \,/\, b=0] \Bigg| \,/\, 2.\end{aligned}$$

On en déduit

$$\mathrm{Adv}^{\text{s-ind}}_{\mathfrak{P},\mathfrak{S}_n}(\mathcal{D},t,q) \geqslant \left|\mathrm{Adv}^{\mathrm{DDH}}_{\mathfrak{P}}(\mathcal{A},t,(q-5)/3) - \mathrm{Adv}^{\mathrm{DDH}}_{\mathfrak{S}_n}(\mathcal{A},t,(q-5)/3)\right| / 2.$$

Et finalement :

$$\mathrm{Adv}^{\mathrm{DDH}}_{\mathfrak{P}}(\mathcal{A},t,(q-5)/3) \leqslant \mathrm{Adv}^{\mathrm{DDH}}_{\mathfrak{S}_n}(\mathcal{A},t,(q-5)/3) + 2\,\mathrm{Adv}^{\text{s-ind}}_{\mathfrak{P},\mathfrak{S}_n}(\mathcal{D},t,q).$$

□

2.5.5 Généralisation et remarques

Les théorèmes 2.3 et 2.4, ainsi que leurs preuves, doivent servir d'illustration des propriétés et des techniques à utiliser sur une famille pseudo-aléatoire de groupes cycliques d'ordre n. En utilisant des méthodes similaires, on peut prouver toute hypothèse « raisonnable » sur une famille pseudo-aléatoire de groupes cycliques.

On peut de plus définir des familles pseudo-aléatoires de groupes cycliques avec couplage d'ordre n, en utilisant deux familles pseudo-aléatoires de permutations : les hypothèses bilinéaires peuvent alors être prouvées dans ce contexte pseudo-aléatoire.

On remarque que dans les preuves des deux théorèmes portant sur les familles pseudo-aléatoires de groupes cycliques, la permutation est fixée au début du jeu dans la réduction : c'est essentiel lorsque la permutation est tirée dans la famille pseudo-aléatoire, et non pas dans la famille de toutes les permutations de $B(n)$. Ainsi, dans ce cadre, on doit impérativement jouer avec une loi de groupe réelle, et non pas répondre aléatoirement à toute requête fraîche, avant de vérifier si la simulation est légitime ou pas. Cela montre la nécessité de rester le plus proche possible d'un modèle concret dans le cadre de la famille générique de groupes cycliques.

Outre cette nécessité de garder un modèle suffisamment flexible, la notion de familles pseudo-aléatoires de groupes cycliques permet d'identifier l'intérêt de certaines requêtes. En effet, on a déjà mentionné la nécessité de tenir compte dans la famille générique de groupes cycliques de l'utilisation de chaînes de bits fraîches, c'est-à-dire d'autoriser un algorithme à soumettre aux oracles des éléments jamais apparus auparavant dans ses requêtes et dans les réponses associées (voir [AF07]). Il est clair que ce type de requêtes n'apporte pas un avantage significatif à un algorithme dans le modèle générique. À l'inverse, dans un modèle pseudo-aléatoire, ce type de requêtes correspond à une attaque de la famille de permutations en tant que famille pseudo-aléatoire de permutations, par opposition à la recherche d'une vulnérabilité dans le problème en lui-même.

Enfin, avec l'utilisation de familles pseudo-aléatoires de groupes, l'avantage d'un algorithme dépend à nouveau (par opposition au cas du modèle générique) de sa puissance de calcul. Cette situation est plus réaliste que celle d'une sécurité basée sur la théorie de l'information.

2.6 Conclusion

Dans ce chapitre, on a réalisé une présentation rigoureuse du modèle générique des groupes, par le biais des définitions de familles génériques de groupes cycliques et de familles de représentations de groupes cycliques. Ce modèle tient le plus possible compte des améliorations au modèle générique initial : la possibilité d'utiliser des chaînes de bits fraîches, l'utilisation des couplages, et la généralisation aux fractions rationnelles...

Dans ce modèle complet, l'étude des collisions est fondamentale, et le résultat du lemme 2.1 donne précisément une borne à l'apparition d'une collision. De cette première borne, on peut déduire des bornes précises de l'avantage d'un algorithme sur divers problèmes liés au logarithme discret, dans un modèle générique : les théorèmes 2.1 et 2.2 en sont deux exemples, sur les problèmes bilinéaire Diffie-Hellman et de q-inversion bilinéaire Diffie-Hellman.

Le modèle de la famille générique de groupes cycliques, avec ou sans couplage, peut être généralisé sous la forme de familles pseudo-aléatoires de groupes. Par réduction, on peut montrer qu'un algorithme sur les problèmes usuels dans une famille pseudo-aléatoire a un avantage maximal relié à celui dans la famille générique et à l'indistinguabilité forte de la famille de permutations sous-jacente.

En fait, justement grâce à cette propriété, le modèle pseudo-aléatoire ne présente pas une amélioration du modèle susceptible d'identifier des propriétés sûres dans le modèle générique, mais non-sûres en réalité. Cependant, de par son plus grand réalisme, ce modèle constitue une amélioration du modèle générique.

Chapitre 3

Diffusion par groupes

La confidentialité dans la diffusion de données vers de nombreux destinataires est un sujet récurrent en cryptographie. Depuis sa formalisation dans [FN93], de nombreuses solutions ont été proposées, pour améliorer l'efficacité des schémas précédents, ou pour couvrir un cadre plus large.

L'objectif de ce chapitre est de présenter la diffusion basée sur des attributs : dans un tel schéma, un émetteur peut exploiter directement une base de données catégorisant les utilisateurs selon divers critères. Il peut ainsi s'adresser uniquement à des utilisateurs vérifiant ou ne vérifiant pas certains critères.

Pour cela, après une présentation de la problématique générale de la diffusion et de quelques schémas emblématiques (partie 3.1), on introduira le concept d'attributs et de groupes d'utilisateurs (partie 3.2). On constatera notamment que les schémas classiques de diffusion ne sont pas adaptés à ce concept, et ne permettent pas une exploitation efficace d'une structure particulière de l'ensemble des utilisateurs. De plus, les schémas actuels de chiffrement basés sur des attributs nécessitent des calculs très coûteux lors d'un déchiffrement.

On propose un modèle pour la définition et la sécurité d'un schéma de diffusion basé sur des groupes d'utilisateurs (partie 3.3), puis un schéma dans ce modèle (partie 3.4) : ce schéma s'appuie fortement sur la structure particulière de l'ensemble des utilisateurs pour diminuer la taille des messages chiffrés, tout en permettant un déchiffrement efficace. La dernière partie sera consacrée à la preuve du schéma (partie 3.5).

3.1 Introduction à la diffusion

Dans de nombreux contextes, un émetteur doit diffuser la même information à de nombreux destinataires. C'est typiquement le cas dans des protocoles de communications réseau, et dans des applications multimédia (radiodiffusion, télédiffusion...) sur divers supports.

La diffusion physique de ces données peut être assurée par de nombreuses méthodes, telles que des protocoles réseau adaptés, l'utilisation d'ondes radio, ou la diffusion via

un réseau câblé ou un satellite... La protection de l'information transmise n'est cependant pas garantie par de tels moyens. Tout d'abord, il est nécessaire de définir les menaces, et les services de sécurité associés pour contrer ces menaces. On peut aisément remarquer qu'une signature asymétrique permet de garantir l'intégrité et l'authentification de l'émetteur pour un coût en stockage, transmissions et calculs assez faible. La confidentialité n'admet par contre pas de réponse aussi simple.

La confidentialité des données transmises nécessite que certains équipements ou certaines personnes doivent pouvoir obtenir l'information distribuée par l'émetteur, alors que d'autres n'y sont pas autorisés. Ainsi, l'émetteur ne souhaite pas diffuser universellement une information, mais la diffuser à un ensemble de destinataires précisément identifiés, et en aucun cas aux autres membres du réseau. Ce cadre correspond à la notion de *multicast* dans le monde des télécommunications, par opposition à la diffusion à l'ensemble des membres d'un réseau, appelé *broadcast*. En cryptologie, dans le cadre d'une protection en confidentialité, la situation est à mi-chemin entre ces deux notions : l'émetteur veut transmettre son message à un ensemble choisi de destinataires, mais le chiffré qu'il construit doit être diffusable à n'importe qui, puisque les canaux de transmission sont considérés par principe comme non sûrs. Le principe est donc fondamentalement celui du *multicast*, alors que sa mise en application peut utiliser un mécanisme de *broadcast*.

3.1.1 Vocabulaire et notations

On appelle désormais **utilisateurs** les équipements ou intervenants du système, quelle que soit leur nature. Le paramètre n correspond au nombre de ces utilisateurs, et on utilisera les entiers de 1 à n pour représenter les utilisateurs. Ainsi, $\mathcal{U} = \{1, \ldots, n\}$ est l'ensemble des utilisateurs.

L'information transmise se base sur une suite de **messages** envoyés successivement vers un ensemble choisi d'utilisateurs, par le biais de **chiffrés** diffusés à tous. Les utilisateurs appartenant à cet ensemble sont dits **privilégiés** pour ce message (ou pour le chiffré associé). À l'inverse, les utilisateurs non-privilégiés sont dits **révoqués**.

Par souci de concision, on appelle simplement **ensemble privilégié** l'ensemble des utilisateurs privilégiés.

3.1.2 Quelques questions

Si la description du problème apparaît jusqu'à présent comme plutôt simple, elle amène immédiatement diverses questions cruciales :

1. Un adversaire est-il un utilisateur révoqué, ou une coalition de tous les utilisateurs révoqués pour un message donné ?
2. Y a-t-il un seul émetteur capable de diffuser des messages vers les utilisateurs, ou n'importe qui doit-il pouvoir le faire ?

3. Un utilisateur est-il capable de mémoriser et d'utiliser durablement des clés calculées à partir de transmissions reçues, ou ne peut-il utiliser que des clés qui ont lui ont été initialement données ?
4. L'ensemble privilégié est-il fixe ou évolue-t-il très significativement, d'un message diffusé au message suivant ?
5. Le nombre des utilisateurs privilégiés est-il petit par rapport à n, ou est-il au contraire proche de n ?

La première question est probablement la plus facile à traiter : si l'utilisation conjointe des moyens cryptographiques de deux utilisateurs révoqués pour un chiffré donné suffisait à obtenir le message associé, cela constituerait une faiblesse probablement inacceptable en pratique. Les premières solutions proposées, dans [FN93] notamment, utilisent un paramètre qui correspond à la taille maximale des coalitions envisagées. Les travaux les plus récents, comme par exemple [BW06], s'efforcent cependant d'apporter une robustesse face à des coalitions de toutes tailles.

La deuxième question est classique en cryptographie, et revient à demander si la fonctionnalité demandée doit être symétrique ou asymétrique. En effet, s'il existe un seul émetteur, on peut « faire confiance » à cet émetteur, et partager des clés secrètes symétriques entre cet émetteur et chacun des utilisateurs. En revanche, si n'importe quel utilisateur ou intervenant extérieur doit pouvoir diffuser des messages, adapter la méthode précédente revient à demander à chaque utilisateur de stocker des clés indépendantes pour chaque émetteur possible, ce qui devient rapidement inconcevable lorsque le nombre de ces émetteurs augmente : il faut donc avoir recours à de la cryptographie asymétrique, afin de mettre à disposition de tous des clés publiques pour diffuser les messages.

Les trois dernières questions conduisent à deux approches radicalement différentes du problème : la première basée sur le partage permanent de clés entre les utilisateurs privilégiés, clés qui évoluent en même temps que l'ensemble privilégié ; l'autre basée sur une distribution initiale de clés de déchiffrement figées suivant une structure particulière.

3.1.2.1 Utilisateurs à mémoire

On appelle **utilisateurs à mémoire** (*stateful receivers*) des utilisateurs capables de mémoriser et d'utiliser pendant de longues périodes de temps des clés qui leur sont transmises via des messages (nécessitant ou non une phase de calcul). Avec de tels utilisateurs, on peut envisager transmettre de manière sécurisée une clé de déchiffrement commune aux utilisateurs privilégiés, et utiliser cette clé pour diffuser des messages très efficacement.

Par contre, il faut renouveler cette clé dès lors qu'un utilisateur jusque là révoqué devient privilégié, ou inversement : lorsque l'ensemble privilégié évolue, il faut générer et distribuer une nouvelle clé à l'ensemble des nouveaux utilisateurs privilégiés.

Une telle stratégie est particulièrement adaptée lorsque l'ensemble privilégié est figé ou évolue faiblement : on peut alors accepter un coût potentiellement élevé lors de modifications de cet ensemble, si l'envoi des messages est très efficace. L'un des inconvénients majeurs de cette approche est que les utilisateurs doivent potentiellement être en permanence en ligne et à l'écoute de tous les messages émis : en effet, les mises à jour des clés sont envoyées ponctuellement, et si un utilisateur « manque » l'une de ces mises à jour, il peut perdre sa capacité à déchiffrer les transmissions émises, même s'il fait toujours partie de l'ensemble privilégié.

Les techniques de renouvellement des clés (et par conséquent les schémas qui en découlent) sont en général plus efficaces lorsque les utilisateurs privilégiés sont peu nombreux.

3.1.2.2 Utilisateurs sans mémoire

On appelle **utilisateurs sans mémoire** (*stateless receivers*) des utilisateurs qui ne peuvent pas faire évoluer leur état interne : ils ne peuvent utiliser à long-terme que des clés qui leur ont été initialement fournies. Cette situation a été définie dans [NNL01], à la suite des travaux présentés dans [KRS99, GSW00] qui proposent des schémas où les états internes des utilisateurs ne sont modifiés qu'après des modifications importantes et durables de l'ensemble privilégié. La distribution initiale des clés joue donc un rôle primordial, et la diffusion doit donc reposer sur une structure particulière de ces clés ; structure qui ne peut évoluer en fonction de l'ensemble privilégié. Cette structure peut porter sur la distribution bien choisie de clés indépendantes, mais également sur le choix de clés corrélées (par le biais de fonctions de dérivation de clés, ou d'équations algébriques) : certains schémas font d'ailleurs appel à ces deux techniques simultanément.

Dans un tel contexte, la diffusion d'un message est plus complexe (et coûteuse) qu'un simple chiffrement. À chaque nouvelle émission, il faut repartir de la structure initialement construite, que l'ensemble des utilisateurs privilégiés ait fortement évolué ou pas. On n'a donc pas de relation de proximité d'un ensemble privilégié par rapport à l'ensemble privilégié qui le suit : on ne peut pas bénéficier d'une quelconque structure commune entre eux, et à l'inverse on ne perd pas en efficacité à devoir changer radicalement d'ensemble privilégié.

On peut néanmoins envoyer efficacement des messages assez longs pour un ensemble privilégié fixe, en utilisant un chiffrement hybride[1] : on considère en effet que même si un utilisateur sans mémoire ne peut stocker sur une période de temps non-définie une clé non fournie initialement, il a quand même la capacité d'enregistrer une clé

[1]Le chiffrement hybride consiste pour l'émetteur à envoyer une clé de déchiffrement et un long message chiffré pour cette clé : un destinataire peut obtenir la clé de déchiffrement et l'utiliser pour obtenir le long message. Cette technique est spécifiquement intéressante quand le chiffrement disponible entre l'émetteur et le destinataire est coûteux (temps de transmission, calculs) : seule une clé est transmise par ce biais, et une autre technique de chiffrement, beaucoup plus efficace, est utilisée pour chiffrer le contenu proprement dit. On utilise en particulier le chiffrement hybride de manière quasi-systématique pour envoyer un contenu long avec de la cryptographie asymétrique.

calculée à partir d'une transmission et de l'utiliser successivement sur les parties d'un long message. En revanche, cette clé ne peut être conservée après la transmission de ce message.

De tels schémas sont donc particulièrement adaptés lorsque l'ensemble privilégié change fréquemment et de manière importante. L'efficacité d'un schéma dépend de la structure choisie des clés initiales. Le cas d'un faible nombre d'utilisateurs révoqués peut par exemple être assez favorable dans certains schémas.

3.1.3 Exemples triviaux

3.1.3.1 Exemple 1 : une clé initiale par ensemble possible

Un premier exemple consiste à associer à chaque ensemble d'utilisateurs une clé (ou un couple de clés) spécifique, et confier initialement à chaque utilisateur l'ensemble des clés de déchiffrement correspondant aux ensembles auxquels il appartient. L'utilisation de la clé de chiffrement correspondant à un ensemble donné pour chiffrer un message assure que les membres de cet ensemble sont exactement les utilisateurs privilégiés pour ce message.

Ce schéma est optimal pour l'envoi des messages, mais le nombre exponentiel de clés (de l'ordre de 2^n, où n est le nombre d'utilisateurs) qui doivent être stockées par les utilisateurs fait qu'il n'est pas envisageable au-delà de quelques utilisateurs.

3.1.3.2 Exemple 2 : des chiffrements indépendants d'une clé commune

Un second exemple consiste à distribuer initialement une clé de déchiffrement spécifique à chaque utilisateur. L'émetteur envoie une clé commune chiffrée indépendamment pour chacun des utilisateurs privilégiés. Tant que l'ensemble privilégié n'évolue pas, l'émetteur utilise cette clé commune pour chiffrer les messages qu'il souhaite envoyer. Lorsque l'ensemble privilégié évolue, l'émetteur génère une nouvelle clé commune et la distribue de la même manière.

Ce schéma, basé sur des utilisateurs avec mémoire, est performant lorsque l'ensemble privilégié n'évolue presque pas, et lorsque le nombre des utilisateurs privilégiés reste faible. En effet, le sur-coût en transmission de ce schéma par rapport au schéma précédent correspond aux phases de distribution des clés qui n'ont lieu que lorsque l'ensemble privilégié change, et le trafic généré lors de l'une de ces distributions est linéaire en le nombre d'utilisateurs privilégiés.

3.1.4 Le schéma LKH (*logical key hierarchy*)

Le schéma LKH (*logical key hierarchy*) a été proposé indépendamment dans [WGL98] et dans [WHA99]. Il s'agit d'une amélioration significative de l'exemple 2 présenté dans le paragraphe précédent, toujours dans le cadre des utilisateurs avec mémoire.

Chacun des utilisateurs est initialement muni, comme dans l'exemple 2, d'une clé qui lui est propre. Par la suite, lorsqu'il devient privilégié, cet utilisateur reçoit un

ensemble de clés. La première est utilisée pour recevoir des messages, alors que les autres servent à renouveler cette première clé lorsque l'ensemble privilégié évolue.

De manière plus concrète, l'ensemble privilégié est muni d'une structure d'arbre, où les utilisateurs privilégiés sont placés sur les feuilles. Un nœud interne correspond à une clé, qui n'est connue que des utilisateurs privilégiés placés sur des feuilles dans le sous-arbre du nœud interne en question. La racine de l'arbre correspond à une clé connue de tous les utilisateurs privilégiés.

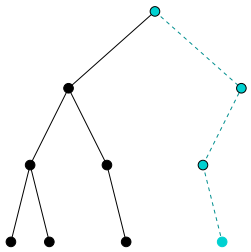

FIG. 3.1 – Schéma LKH : utilisateur privilégié et ses clés connues

Lorsque l'émetteur ajoute un utilisateur privilégié, il le place sur une nouvelle feuille de l'arbre. L'émetteur renouvelle les clés correspondant aux nœuds « parents » de cette feuille, jusqu'à la racine : pour chacun de ces nœuds, il génère une nouvelle clé et l'envoie chiffrée par l'ancienne clé correspondant à ce nœud. Le nouvel utilisateur privilégié reçoit séparément toutes ces clés, chiffrées par sa clé personnelle.

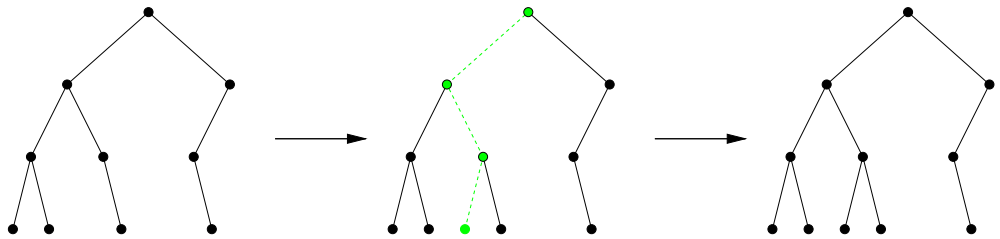

FIG. 3.2 – Schéma LKH : ajout d'un utilisateur privilégié

Lorsque l'émetteur révoque un utilisateur privilégié, il enlève la feuille correspondante de l'arbre, et renouvelle les clés correspondant aux nœuds « parents » de cette feuille, jusqu'à la racine : les clés sont renouvelées du bas vers le haut de l'arbre, chacune des nouvelles clés est envoyée chiffrée par les clés correspondant à ses descendants directs dans l'arbre.

Lors de l'ajout ou de la révocation d'un utilisateur privilégié, il faut donc envoyer un nombre de clés chiffrées logarithmique en la taille de l'ensemble privilégié. Des améliorations ont été apportées à ce schéma dans [CGI+99, CMN99, PST01], notamment par l'utilisation de diverses fonctions de dérivation de clés pour permettre des envois plus réduits de clés chiffrées.

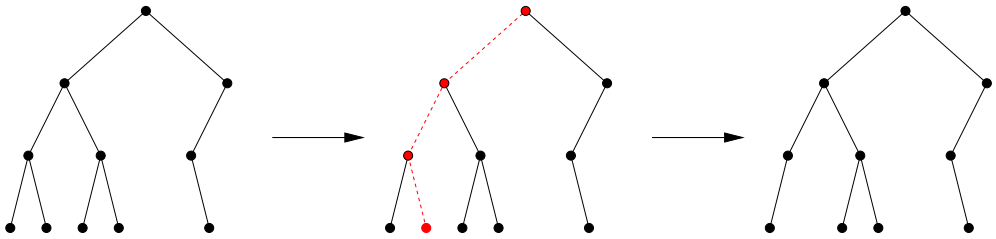

FIG. 3.3 – Schéma LKH : révocation d'un utilisateur privilégié

3.1.5 Les schémas de recouvrement par sous-ensembles

Le principe de recouvrement par des sous-ensembles fixes (*subset cover*) a été proposé dans [NNL01] pour construire des schémas pour des utilisateurs sans mémoire. Il s'agit en particulier de disposer de schémas efficaces dans le cas où le nombre d'utilisateurs révoqués est faible.

3.1.5.1 Principe

L'initialisation du système consiste à définir une famille $\mathcal{F}$ d'ensembles particuliers d'utilisateurs. Des clés sont générées indépendamment pour chacun de ces ensembles, et un utilisateur obtient initialement les clés de déchiffrement correspondant aux ensembles auxquels il appartient.

Pour distribuer un message, un émetteur exprime l'ensemble privilégié choisi comme la réunion d'ensembles de la famille $\mathcal{F}$ initialement définie. Il lui suffit alors de chiffrer son message par chacune des clés de chiffrement correspondant à ces ensembles.

Le choix initial de la famille $\mathcal{F}$ doit donc permettre d'exprimer n'importe quel ensemble d'utilisateurs comme réunion d'ensembles appartenant à $\mathcal{F}$, et le calcul des ensembles impliqués dans cette réunion doit être facile. À partir de cette contrainte, il faut trouver un bon compromis :

- du point de vue d'un utilisateur, le nombre d'ensembles de la famille $\mathcal{F}$ auxquels il appartient doit être petit, car il doit mémoriser une clé de déchiffrement spécifique pour chacun de ces ensembles ;
- du point de vue d'un émetteur, la famille $\mathcal{F}$ doit permettre d'exprimer n'importe quel ensemble privilégié en la réunion d'un faible nombre d'ensembles appartenant à cette famille.

On peut d'ailleurs voir les deux exemples triviaux (partie 3.1.3) comme les deux situations extrêmes de ce choix de famille. Dans l'exemple 1, la famille contient tous les sous-ensembles possibles de $\mathcal{U}$ (cas le plus favorable à l'émetteur, qui n'utilise qu'une clé quel que soit l'ensemble privilégié choisi). Dans une version « sans mémoire » de l'exemple 2, au contraire, la famille contient tous les singletons de $\mathcal{U}$ (cas le plus favorable aux utilisateurs, qui ne mémorisent qu'une seule clé).

3.1.5.2 Le schéma CS (*complete subtree*)

Le schéma CS (*complete subtree*) est le premier schéma proposé dans [NNL01]. Il repose sur l'utilisation d'un arbre binaire d'une manière proche du schéma LKH (§3.1.4). La différence essentielle est que tous les utilisateurs sont dans l'arbre, et pas seulement les utilisateurs privilégiés à un instant donné. Ainsi, l'arbre n'évolue pas :

- tous les utilisateurs du système sont placés sur les feuilles de l'arbre, et disposent d'une clé de déchiffrement spécifique ;
- un nœud interne correspond à une clé de déchiffrement qui n'est connue que des utilisateurs placés sur les feuilles du sous-arbre de ce nœud interne.

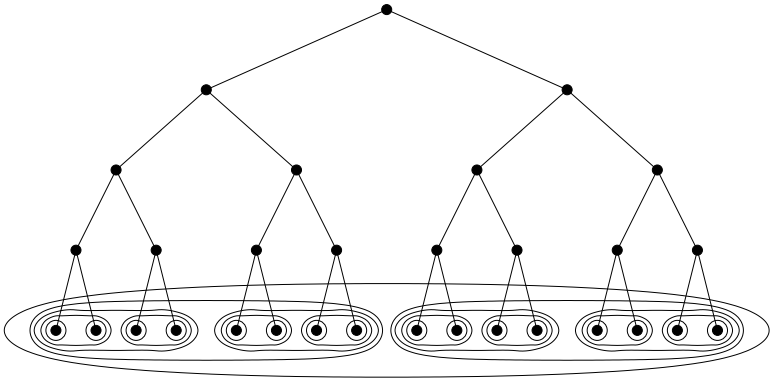

FIG. 3.4 – Schéma CS : famille de sous-ensembles

La famille de sous-ensembles qui correspond à cette structure est donc l'ensemble des descendants des nœuds de l'arbre, représentée en figure 3.4 :

$$\mathcal{F}_{\mathrm{CS}} = \{\mathrm{desc}(x) \,/\, x \text{ nœud de l'arbre}\}.$$

Pour déterminer les ensembles à utiliser dans une diffusion, un émetteur détermine les sous-arbres les plus grands contenant exclusivement des utilisateurs privilégiés. La réunion des ensembles correspondant aux racines de ces sous-arbres est égale à l'ensemble privilégié.

La figure 3.5 représente les cinq nœuds et ensembles utilisés pour recouvrir l'ensemble privilégié composé des utilisateurs A, B, E, F, G, H, I, J, K et N.

Dans le schéma CS, les utilisateurs doivent stocker un nombre de clés de déchiffrement logarithmique en n. Le nombre de clés de chiffrement utilisées pour diffuser un message est borné par $n-r$ et par $r\log(n/r)$, où r est le nombre d'utilisateurs révoqués. Lorsque le nombre d'utilisateurs révoqués n'est ni petit ni proche de n, le nombre de clés à utiliser peut varier fortement en fonction de la répartition des utilisateurs privilégiés dans l'arbre.

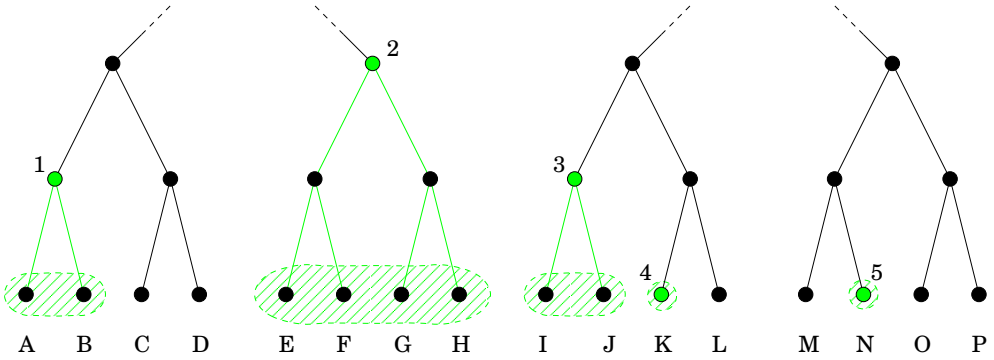

FIG. 3.5 – Schéma CS : recouvrement de l'ensemble privilégié

3.1.5.3 Le schéma SD (*subset difference*)

Le schéma SD (*subset difference*) a également été proposé dans [NNL01] : il utilise la même structure en arbre binaire que le schéma CS, mais la famille de sous-ensembles est différente. On définit un ensemble de la famille comme la différence de deux ensembles de la famille $\mathcal{F}_{\mathrm{CS}}$ inclus l'un dans l'autre. On ajoute seulement l'ensemble de tous les utilisateurs, ce qui donne la définition formelle suivante :

$$\begin{aligned}\mathcal{F}_{\mathrm{SD}} &= \{\mathcal{U}\} \cup \{(S_1 \setminus S_2) \ / \ (S_1, S_2) \in {\mathcal{F}_{\mathrm{CS}}}^2,\ S_2 \subset S_1\},\\ &= \{\mathcal{U}\} \cup \{\mathrm{desc}(x_1) \setminus \mathrm{desc}(x_2) \ / \ x_2 \in \mathrm{desc}(x_1)\}.\end{aligned}$$

Cette famille $\mathcal{F}_{\mathrm{SD}}$ contient en particulier la famille $\mathcal{F}_{\mathrm{CS}}$ précédemment étudiée, puisque les descendants d'un nœud x peuvent s'écrire comme les descendants du « père » de x, exceptés les descendants du « frère » de x (seule la racine est traitée séparément). Cette famille permet donc logiquement de recouvrir les ensembles privilégiés par un nombre plus petit d'ensembles appartenant à la famille.

Pour déterminer le recouvrement d'un ensemble privilégié par des ensembles appartenant à $\mathcal{F}_{\mathrm{SD}}$, un émetteur doit d'abord identifier les sous-arbres contenant exclusivement des utilisateurs révoqués. Il utilise les clés correspondant aux sous-arbres les plus grands contenant des utilisateurs privilégiés et au plus l'un des sous-arbres révoqués préalablement identifiés.

La figure 3.6 représente les trois couples de nœuds et ensembles utilisés pour recouvrir le même ensemble privilégié que dans la figure 3.5 : l'efficacité de la diffusion est donc meilleure que dans le schéma CS.

Plus précisément, le nombre de clés de chiffrement utilisées pour diffuser un message est borné par $n-r$ et par $2r-1$, où r est le nombre d'utilisateurs révoqués. Dans le cas où r est petit, on gagne un facteur logarithmique en r par rapport au schéma précédent. Cependant, de la même manière que pour le schéma CS, lorsque ces deux bornes sont élevées, il existe des situations favorables où le nombre de clés à utiliser est très petit, et des situations très défavorables, proches des bornes.

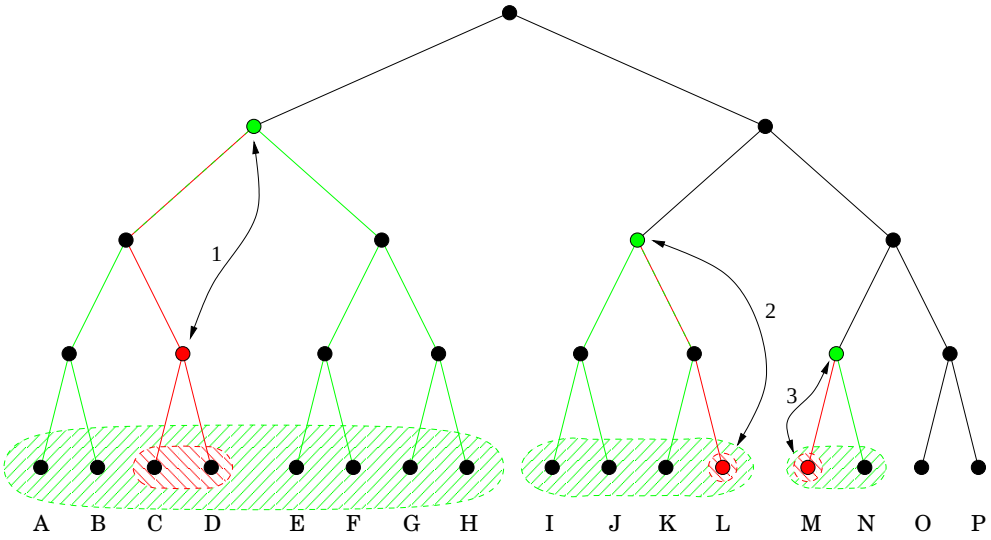

FIG. 3.6 – Schéma SD : recouvrement de l'ensemble privilégié

La contrepartie est que les utilisateurs doivent a priori stocker un nombre de clés de déchiffrement bien plus grand, linéaire en n. Mais par l'utilisation de techniques de dérivation de clés, ce stockage peut être ramené à un nombre de clés proportionnel à $\log(n)^2$, d'où n'importe laquelle des clés de déchiffrement nécessaire peut être calculée en un nombre d'opérations logarithmique en n.

Par contre, dans un cadre asymétrique, les clés publiques correspondant à toutes les clés de déchiffrement doivent être connues par un émetteur, ce qui représente un stockage proportionnel à $n\log(n)$. Ce problème a été résolu dans [DF02], grâce à la cryptographie à base de couplage : le stockage nécessaire au chiffrement est réduit à une taille constante, à la condition de disposer d'un schéma de chiffrement basé sur l'identité hiérarchique, avec des chiffrés de taille constante. Un tel schéma a été proposé quelques années plus tard dans [BBG05].

Le schéma SD a été amélioré par la suite : le schéma LSD (*layered subset difference*) proposé dans [HS02] élimine un grand nombre de clés, en remarquant qu'on pouvait exprimer des ensembles de la famille $\mathcal{F}_{\mathrm{SD}}$ comme une réunion d'un petit nombre d'autres éléments de cette famille. Le schéma SSD (*stratified subset difference*) proposé dans [GST04] améliore encore ces performances, pour atteindre un stockage par utilisateur proportionnel à $\log(n)$ tout en conservant le même ordre de grandeur pour le nombre de clés utilisés dans une diffusion.

3.1.6 Une dépendance forte en r ou $n - r$

Tous les schémas présentés jusqu'à présent ont un inconvénient majeur : dans un contexte où l'ensemble privilégié est fortement variable, et lorsque r et $n - r$ ne sont pas spécialement petits, leur utilisation est inenvisageable :

– les schémas de type LKH sont extrêmement coûteux en raison des variations de l'ensemble privilégié, et il faut reconstruire toute la structure des clés partagées lors de chaque envoi d'un nouveau chiffré, ce qui nécessite des transmissions au moins linéaires en $n - r$;
– les schémas de type CS ou SD présentent des chiffrés de taille généralement linéaire en r ou $n - r$.

Il n'est pas vraiment surprenant que cette situation soit coûteuse : le chiffré et les éventuelles transmissions nécessaires à l'établissement d'une clé commune contiennent la description de l'ensemble privilégié. Or cette description nécessite en toute généralité n bits d'information (un bit pour chaque utilisateur), qui peut être ramené à $r \log(n)$ ou $(n-r)\log(n)$ dans des cas favorables, en optant pour une liste des utilisateurs révoqués ou une liste des utilisateurs privilégiés.

Par contre, dans des applications concrètes, l'ensemble privilégié n'est pas quelconque et s'appuie sur des groupes particuliers d'utilisateurs ayant des profils proches. Sa description peut être largement compactée, en utilisant ces groupes d'utilisateurs, et la borne liée à la théorie de l'information ne tient plus. L'objectif du reste de ce chapitre et d'étudier ce type de cas et de proposer une solution efficace de diffusion.

3.2 Groupes d'utilisateurs définis par des attributs

3.2.1 Problématique des groupes d'utilisateurs

L'émetteur souhaite diffuser son message à un ensemble d'utilisateurs correspondant à des critères précis, prédéfinis, à l'exclusion d'utilisateurs clairement identifiés en faible nombre, identifiés après la distribution des clés de déchiffrement.

On considère l'exemple d'une diffusion payante de programmes télévisés. Lors de la diffusion des programmes, l'émetteur souhaite naturellement en limiter l'accès aux utilisateurs dont l'abonnement est en cours de validité. Mais il peut exister de très nombreux critères supplémentaires permettant de choisir ou d'écarter des utilisateurs : type d'abonnement, localisation, date de naissance de l'utilisateur principal, limitation d'accès décidée par l'utilisateur... La figure 3.7 montre un exemple de structure d'accès utilisable par un émetteur.

Id	Nom	Né le	Abonnement	Fin	Région
1	Alice	31/01/81	Cinéma	Déc 2007	Île de France
2	Bob	01/02/81	Actualité/Sports	Août 2009	Bretagne
3	Charlie	19/05/77	Divertissements	Mai 2008	Île de France
4	Dan	20/09/83	Actualité/Cinéma	Mai 2008	Centre
5	Eve	24/01/79	Divertissements/Sports	Jan 2008	Rhône-Alpes

FIG. 3.7 – Critères de diffusion

L'émetteur correspondant à cette structure peut envisager la diffusion d'un film (catégorie cinéma), en janvier 2008, avec une offre commerciale permettant en plus

aux abonnés hors Île de France nés en janvier de voir ce film : seuls Dan et Eve sont autorisés à voir ce film.

On peut supposer que l'émetteur connaît avant la distribution initiale des clés de déchiffrement les critères qu'il souhaite utiliser pour caractériser ses diffusions. Dans le pire des cas, il peut prendre en compte de nombreux critères, ne sachant pas précisément lesquels il utilisera ultérieurement. Par contre, certains utilisateurs doivent être révoqués suite à un usage frauduleux de leurs clés de déchiffrement. Ils ne peuvent être connus avant la mise en place du système, mais ils sont a priori peu nombreux.

3.2.2 Schémas préexistants dans ce contexte

Le schéma LKH (§3.1.4) est mal adapté à cette situation particulière : des ensembles privilégiés successifs définis à partir de critères proches peuvent avoir des différences de composition très importantes. Cela représente donc un grand nombre d'ajouts et de révocations d'utilisateurs, qui sont coûteuses dans ce schéma.

Les schémas basés sur les recouvrements d'ensembles (§3.1.5) proposés dans [NNL01, HS02, GST04] sont eux plus favorables, même si les bornes mesurant l'efficacité d'une diffusion ne sont pas bonnes dans ce contexte. La structure en arbre est choisie librement par l'émetteur, et il peut utiliser cette liberté pour créer un arbre qui correspond aux critères pertinents : la subdivision en sous-arbres correspond aux différents attributs des utilisateurs. Ainsi, en reprenant l'arbre à 16 utilisateurs mentionné en figures 3.5 et 3.6, les utilisateurs A à H ont le même premier attribut, les utilisateurs A à D et I à L ont le même second attribut, etc.

Avec une telle structure d'arbre calquée sur les critères de diffusion, les messages basés sur les critères utilisés dans la partie haute de l'arbre sont efficaces (voir figure 3.8). Les critères utilisés dans la partie basse de l'arbre sont au contraire très coûteux lors de leur utilisation, et correspondent même aux pires cas envisageables sur de telles structures (voir figure 3.9).

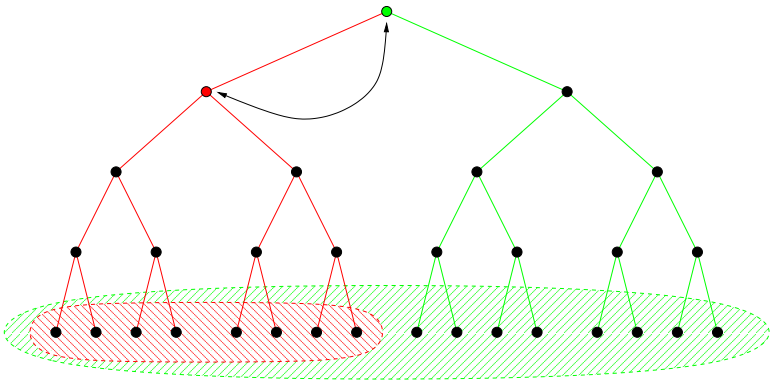

FIG. 3.8 – Schéma SD : révocation du premier critère associé à l'arbre

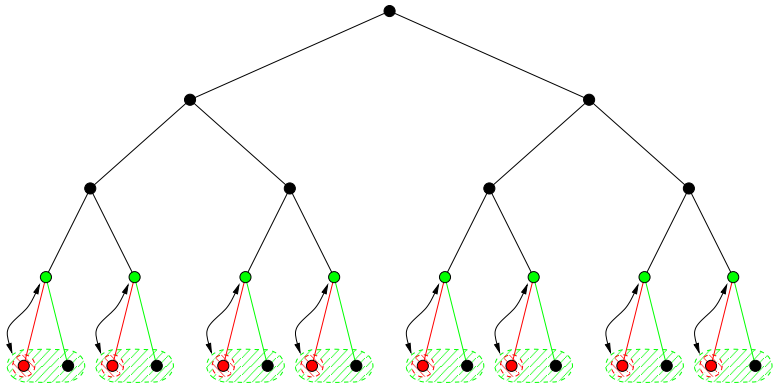

FIG. 3.9 – Schéma SD : révocation du dernier critère associé à l'arbre

En outre, la structure en arbre équilibré (binaire de surcroît sur certains schémas) implique une bonne répartition des utilisateurs sur l'ensemble des critères, et des critères définissant le même nombre de classes. Ces deux contraintes sont assez fortes, et plutôt irréalistes.

Un schéma à clé publique pour des utilisateurs sans mémoire proposé plus récemment dans [BGW05] (leur premier schéma) est décrit comme ayant des clés privées et des messages chiffrés de tailles constantes, c'est-à-dire indépendantes du nombre n total d'utilisateurs et du nombre r d'utilisateurs révoqués. En fait, dans ce schéma, la connaissance de la clé publique du système est nécessaire aux utilisateurs pour déchiffrer, en plus de leur clé privée. Cette clé publique étant linéaire en n, le stockage résultant pour les utilisateurs est donc également linéaire en n. De plus, une description de l'ensemble privilégié est nécessaire dans chaque diffusion, ce qui ne permet pas d'assurer des messages chiffrés de taille constante.

Une dernière possibilité serait d'utiliser un schéma de chiffrement hiérarchique basé sur l'identité avec des « jokers » (*wildcards*), comme présenté dans [ACD+06, BDNS06], dans l'adaptation suggérée dans [DF02] des schémas de recouvrement par des sous-ensembles. La sélection des utilisateurs correspondant à certains attributs devient alors envisageable, en utilisant un schéma de type CS, mais une application à un schéma de type SD semble plus délicate. Enfin, les schémas de chiffrement hiérarchique basé sur l'identité avec des « jokers » actuellement connus ne sont pas à taille de chiffré constante, ce qui signifie une perte très significative d'efficacité dans l'adaptation à la diffusion, comme mentionné dans [BBG05].

Les schémas traditionnels ne sont donc manifestement pas adaptés au contexte des groupes d'utilisateurs, qui semble pourtant très intéressant en pratique.

3.2.3 Objectif

L'objectif est donc de disposer d'un schéma de diffusion à clé publique, sûr contre des coalitions de toutes tailles et efficace lorsque l'ensemble privilégié associé à un

message chiffré est défini en fonction de critères vérifiés par les utilisateurs.

Le fait que le schéma soit à clé publique permet à de nombreux émetteurs de partager le même réseau de distribution de contenus. De plus, elle permet une séparation claire des rôles entre un gestionnaire du réseau (qui s'occupe de la distribution des clés aux utilisateurs, de l'infrastructure de diffusion) et des fournisseurs de contenus (qui souhaitent n'avoir chacun à faire confiance qu'au gestionnaire du réseau, et pas aux autres fournisseurs de contenus partageant le même réseau).

Pour chaque **critère** initialement choisi, chaque utilisateur appartient à une classe particulière d'utilisateurs, définie par un **attribut** commun : cette classe d'utilisateurs est appelée **groupe d'utilisateurs**. Par exemple, pour le critère « mois de naissance » dans la base définie par la figure 3.7, il existe 4 groupes d'utilisateurs (Alice et Eve / Bob / Charlie / Dan) définis par 4 attributs différentes (janvier / février / mai / septembre).

Lorsque l'ensemble privilégié d'un message est défini en fonction de l'appartenance des utilisateurs aux groupes, on souhaite que le nombre de clés de chiffrement utilisées par l'émetteur ne dépende que du nombre de groupes impliqués, et pas du nombre d'utilisateurs privilégiés ou révoqués. Ainsi, la révocation d'un groupe d'utilisateurs défini par un unique attribut commun (« fin d'abonnement en 2007 ») doit pouvoir être réalisée par un chiffré de taille constante.

De plus, la gestion des groupes doit être possible quelle que soit la structure de ces groupes, à la différence des schémas CS ou SD qui imposent des contraintes fortes, comme mentionné précédemment (§3.2.2).

3.2.4 Une solution coûteuse

En fait, des solutions de chiffrement basé sur des attributs existent déjà. Un schéma restreint à un unique attribut a été proposé dans [SW05], puis généralisé dans [GPSW06]. Ces schémas sont dits à politique d'accès basée sur les clés (*key-policy*) : pour chaque utilisateur, une politique d'accès est définie, et une clé de déchiffrement est générée en fonction de cette politique d'accès. Lors du chiffrement d'un message, des attributs sont choisis et utilisés. Si ces attributs vérifient la politique d'accès d'un utilisateur, alors la clé de déchiffrement de cet utilisateur lui permettra de déchiffrer. À l'inverse, toute coalition d'utilisateurs correspondant à des politiques d'accès invalides n'est pas en mesure de déchiffrer.

Une variation du schéma [GPSW06] a été proposée dans [BSW07] : il s'agit d'inverser le rôle de la clé de déchiffrement et du chiffré : les attributs sont définis et utilisés pour générer les clés de déchiffrement, tandis que la politique d'accès est choisie et utilisée dans le chiffrement. On parle alors de politique d'accès basée sur les chiffrés (*ciphertext-policy*). Dans ce contexte, un schéma peut être facilement utilisé pour faire de la diffusion : non seulement la politique d'accès permet de décrire facilement un ensemble privilégié, mais en plus, avec un choix adapté des attributs pour les utilisateurs, n'importe quel ensemble privilégié peut être décrit.

Ces schémas reposent sur des techniques de partage de secret, et plus particulièrement celle de Shamir [Sha79] basée sur des polynômes. Une politique d'accès est définie par une formule logique dont les événements élémentaires correspondent à la présence d'un attribut. Les opérations permises sont des fonctions à seuil : en particulier, les opérations ET et OU sont exploitables. Par contre, avec une telle structure, la révocation est difficile, puisqu'on ne dispose pas de l'événement élémentaire correspondant à l'absence d'un attribut, ni de la négation dans les opérations permises. Ce problème est résolu dans [OSW07] qui généralise la structure d'accès de [GPSW06] pour permettre l'utilisation d'opérations de négation.

En combinant les résultats de [BSW07] et de [OSW07], on obtient ainsi un schéma qui répond à l'objectif décrit précédemment (section 3.2.3). Par contre, ce schéma nécessite des calculs importants lors du déchiffrement : le schéma repose sur des groupes avec couplage, et le nombre de couplages à calculer lors d'un déchiffrement est linéaire en le nombre d'attributs utilisés dans la politique d'accès. Or dans la diffusion, le déchiffrement doit être réalisé par les utilisateurs, c'est-à-dire par des décodeurs dont on cherche à minimiser le coût. On ne peut donc pas accepter des calculs aussi importants dans la phase de déchiffrement.

3.2.5 Contribution

Dans ce chapitre, on présente un nouveau schéma de diffusion avec gestion de groupes d'utilisateurs, dont la conception repose sur des mécanismes complètement différents des schémas [GPSW06, BSW07, OSW07]. Dans ce schéma, le déchiffrement nécessite un nombre fixe (3) de couplages, quel que soit le nombre d'attributs associés à un utilisateur et quelle que soit la politique d'accès utilisée lors du chiffrement.

Ce schéma nécessite en revanche l'utilisation de politiques d'accès d'une forme particulière : il s'agit exclusivement de disjonctions (opérations OU, s'appuyant sur le principe de recouvrement par des ensembles de [NNL01]) de conjonctions (opérations ET) d'événements élémentaires correspondant à la présence ou à l'absence d'un attribut. De telles politiques d'accès sont assez restrictives, mais en pratique, elles correspondent relativement bien à ce que l'on attend d'une gestion des attributs dans un protocole de diffusion.

Le principe de ce schéma repose sur la théorie des polynômes, et en particulier sur le calcul de plus grands diviseurs communs. En effet, une clé de déchiffrement est associée à un polynôme dont les racines dépendent des attributs correspondant à l'utilisateur. Le chiffrement est associé à un second polynôme, dont les racines dépendent des attributs révoqués et des attributs requis. Le déchiffrement consiste à calculer le plus grand commun diviseur commun de ces polynômes. Pour tout polynôme associé à une clé valide pour un chiffré donné, ce diviseur est exactement le polynôme dont les racines correspondent aux attributs requis, indépendamment de la clé de déchiffrement utilisée. Pour tout polynôme associé à une clé non-valide, ce diviseur est différent.

En termes de tailles de clés et de chiffrés, la taille d'une clé de déchiffrement est linéaire en le nombre d'attributs correspondant à cette clé. La taille d'un chiffré est

linéaire en le nombre d'attributs utilisés dans la politique d'accès. La situation est donc particulièrement favorable dans le cadre de la diffusion. En revanche, la clé de chiffrement est linéaire en le nombre total d'attributs du système, donc très longue. Ce n'est cependant pas un obstacle majeur, puisqu'un émetteur doit disposer d'une base de données avec tous les utilisateurs du système et leurs attributs. De plus, un émetteur qui n'a besoin de manipuler qu'un petit nombre d'attributs peut utiliser une clé de chiffrement linéaire en ce petit nombre d'attributs.

En termes de puissance de calcul nécessaire, le chiffrement nécessite un unique couplage et un nombre d'opérations de groupe (additions et multiplications par des scalaires) linéaire en le nombre d'attributs utilisés dans la politique d'accès choisie. De plus, si le couplage est considéré comme une opération trop coûteuse pour le chiffrement, on peut l'éviter en rallongeant la clé publique (via l'insertion d'éléments du groupe d'arrivée du couplage) et au prix de nouvelles opérations de groupe. Le déchiffrement nécessite essentiellement trois couplages, en plus des opérations de groupe classiques dont le nombre est linéaire en le nombre d'attributs utilisés dans la politique d'accès et le nombre d'attributs associés à la clé de déchiffrement.

3.3 Modèles et sécurité

3.3.1 Groupes d'utilisateurs

Dans les applications visées, le nombre d'utilisateurs est très grand, et le nombre de groupes d'utilisateurs sera également très grand (il est souhaitable de disposer pour chaque utilisateur d'un groupe contenant uniquement cet utilisateur, voir §3.3.3). Par contre, un utilisateur n'appartient qu'à un nombre restreint de groupes. On choisit donc un formalisme des groupes d'utilisateurs adapté.

Définition 3.1 (Groupe d'utilisateurs) *Un groupe d'utilisateurs est une partie $\mathcal{G}$ de $\mathcal{U} = \{1, \ldots, n\}$. On considère un ensemble de l groupes d'utilisateurs $(\mathcal{G}_i)_{1 \leqslant i \leqslant l}$; on associe inversement à tout utilisateur $u \in \mathcal{U}$ l'ensemble $\mathcal{B}(u)$ des groupes auxquels il appartient : $\mathcal{B}(u) = \{i \in \{1, \ldots, l\} \,/\, u \in \mathcal{G}_i\} \subset \{1, \ldots, l\}$.*

3.3.2 Diffusion par groupes

Le modèle suivant de la diffusion avec gestion de groupes s'appuie sur les définitions données dans [BGW05] (diffusion) et dans [BSW07] (attributs ou groupes d'utilisateurs). Il est cependant adapté spécifiquement aux structures d'accès considérées par le schéma étudié, défini en partie 3.4.

Il ne prend pas en compte le « dynamisme » possible des schémas, c'est-à-dire la capacité d'un schéma de diffusion par groupes à autoriser de nouveaux utilisateurs à rejoindre le système progressivement, sans modifier les clés de déchiffrement déjà données et utilisées. Cette notion de dynamisme a été suggérée dans [DPP07], et un modèle adapté a été conçu. Le schéma construit plus loin semble cependant vérifier une telle propriété de dynamisme.

Définition 3.2 (Schéma de diffusion avec gestion de groupes) *Un schéma de diffusion à clé publique avec gestion de groupes est un ensemble de trois algorithmes, définis à partir d'un paramètre de sécurité λ :*

- ***Setup***$(\lambda, n, (\mathcal{B}(u))_{1\leqslant u\leqslant n})$ *: algorithme randomisé de génération des clés, qui prend pour entrées le paramètre de sécurité λ, le nombre d'utilisateurs n et une description $(\mathcal{B}(u))_{1\leqslant u\leqslant n}$ de groupes d'utilisateurs. Il retourne une clé publique de chiffrement* EK*, et n clés de déchiffrement* $(\mathrm{dk}(u))_{1\leqslant u\leqslant n}$ *;*
- ***Encrypt***$(\mathrm{EK}, \mathcal{B}_N, \mathcal{B}_R)$ *: algorithme randomisé de chiffrement, qui prend en entrée une clé publique de chiffrement* EK *et deux ensembles de groupes, $\mathcal{B}_N$ et $\mathcal{B}_R$. Il retourne un entête* hdr *et une clé K de chiffrement de message, appartenant à un ensemble fini de clés $\mathcal{K}$;*
- ***Decrypt***$(\mathrm{dk}(u), \mathrm{hdr})$ *: algorithme déterministe de déchiffrement, qui prend en entrée une clé de déchiffrement* $\mathrm{dk}(u)$ *donnée à un utilisateur u et un entête* hdr*. Si l'entête est issu d'un appel à l'algorithme de chiffrement utilisant $(\mathcal{B}_N, \mathcal{B}_R)$ tels que $\mathcal{B}_N \subset \mathcal{B}(u)$ et $\mathcal{B}(u) \cap \mathcal{B}_R = \emptyset$, alors cet algorithme retourne une clé de chiffrement de message $K \in \mathcal{K}$.*

Dans le mécanisme de chiffrement, un message M est chiffré avec une clé de chiffrement de message K, et le chiffré résultant est diffusé avec un entête hdr. Les utilisateurs privilégiés pour ce message :

- appartiennent simultanément à tous les groupes « requis », définis par $\mathcal{B}_N$;
- n'appartiennent à aucun des groupes « révoqués », définis par $\mathcal{B}_R$.

Ces utilisateurs privilégiés sont capables d'obtenir la clé de chiffrement de messages K, et d'en déduire le message M à partir de son chiffré : il s'agit d'un mécanisme de chiffrement hybride.

On remarque que dans les définitions précédentes, la clé de déchiffrement et l'entête sont les seuls éléments nécessaires à un utilisateur privilégié pour déterminer la clé de chiffrement de messages K : la connaissance de la clé publique de chiffrement du schéma, d'autres éléments publics, ou de l'ensemble privilégié ne sont pas nécessaires pour réaliser le déchiffrement. Le coût en transmission d'un tel schéma de diffusion correspond donc à la taille de l'entête.

En fait, dans le schéma défini ultérieurement (partie 3.4), la connaissance de l'ensemble privilégié est implicitement transmise dans l'entête, qui contient des éléments représentant les groupes requis et les groupes révoqués.

3.3.3 Choix des groupes

Dans cette définition, le chiffrement n'est pas basé sur un ensemble privilégié quelconque, mais sur la description d'un ensemble privilégié à partir des groupes d'utilisateurs. Il n'est donc pas garanti qu'on puisse diffuser un message à n'importe quel ensemble privilégié, alors que c'est traditionnellement ce que l'on attend d'un schéma de diffusion. On s'écarte donc temporairement de la description du modèle de sécurité pour traiter plus précisément le choix des groupes d'utilisateurs.

Tout d'abord, on envisage un schéma décrit par la définition 3.2 de la section précédente comme un schéma de recouvrement par sous-ensembles (décrit en §3.1.5) : on s'autorise donc à utiliser plusieurs chiffrés « élémentaires », c'est-à-dire décrits par la définition, pour envoyer le même message ou une même clé lorsqu'on utilise un chiffrement hybride pour plus d'efficacité (voir §3.1.2.2). Pour permettre une diffusion à n'importe quel ensemble privilégié, il suffit donc de pouvoir exprimer un tel ensemble privilégié comme une réunion d'ensembles correspondant aux destinataires de chiffrés élémentaires.

Une condition très simple permet de garantir que n'importe quel ensemble privilégié est possible : il suffit que tout utilisateur soit l'unique membre d'un groupe. Dans cette hypothèse, on peut créer des messages chiffrés à destination d'un unique utilisateur (en imposant l'appartenance au groupe dont il est l'unique membre) ou à destination de tous les utilisateurs sauf un (il suffit de révoquer le groupe dont il est l'unique membre). Ainsi, on peut envoyer un message « séparément » à tous les membres d'un ensemble privilégié quelconque, en utilisant un chiffré élémentaire par utilisateur privilégié. Donc en plus des groupes déterminés par les attributs des utilisateurs, on ajoute un groupe par utilisateur, qui ne contient que cet utilisateur.

En fait, l'objectif recherché n'est pas prioritairement de pouvoir diffuser un message à n'importe quel ensemble privilégié, mais il semble raisonnable de permettre à un émetteur de pouvoir le faire avec la meilleure efficacité possible. Pour cela, il suffit de créer encore de nouveaux groupes, correspondant à une structure en arbre binaire des utilisateurs : pour chaque noeud interne, on définit un nouveau groupe, contenant tous ses descendants dans l'arbre. On obtient alors exactement les chiffrés élémentaires correspondant à la méthode SD (voir §3.1.5.3) avec une sélection d'un groupe et une révocation d'un groupe. Si les chiffrés élémentaires ont une taille fixe dans ces conditions (ce qui est le cas avec le schéma présenté ultérieurement), on obtient un coût en transmission équivalent à la méthode SD lorsque l'ensemble privilégié est quelconque.

3.3.4 Modèle de sécurité

On considère la sécurité sémantique d'un schéma de diffusion à clé publique, avec gestion de groupes, défini comme précédemment.

Définition 3.3 (Jeu IND-CPA) *Le jeu d'indistinguabilité pour des attaques à clairs choisis, noté IND-CPA, pour un schéma de diffusion se déroule entre un challenger, qui modélise la génération des clés et la diffusion des messages, et un adversaire, c'est-à-dire un algorithme randomisé :*

- *le challenger et l'adversaire sont initialisés avec une description d'un ensemble de l groupes d'utilisateurs : $(\mathcal{B}(u))_{1\leqslant u\leqslant n}$;*
- *l'adversaire $\mathcal{A}$ détermine deux ensembles de groupes, $\mathcal{B}_N$ et $\mathcal{B}_R$, correspondant au chiffrement sur lequel il veut être défié ;*
- *le challenger génère les clés par l'appel $\textbf{\textit{Setup}}(\lambda, n, (\mathcal{B}(u))_{1\leqslant u\leqslant n})$, et transmets à l'adversaire $\mathcal{A}$ la clé publique de chiffrement,* EK*, ainsi que les clés de déchif-*

frement $\mathrm{dk}(u)$ *des utilisateurs* u *révoqués par l'utilisation des ensembles de groupes choisis précédemment, c'est-à-dire tels que :* $\mathcal{B}_N \cap \mathcal{B}(u) \neq \mathcal{B}_N$ *ou* $\mathcal{B}_R \cap \mathcal{B}(u) \neq \emptyset$ *;*
- *le challenger lance* ***Encrypt***$(\mathrm{EK}, \mathcal{B}_N, \mathcal{B}_R)$*, et obtient un entête* hdr *et une clé de chiffrement de message* $K \in \mathcal{K}$*. Il choisit aléatoirement et uniformément un bit* b *dans* $\{0,1\}$*, puis une clé de chiffrement de message* K_{1-b} *dans* $\mathcal{K}$*. Il fixe* $K_b = K$*, et divulgue le triplet* (hdr, K_0, K_1) *à l'adversaire* $\mathcal{A}$ *;*
- *l'adversaire* $\mathcal{A}$ *répond par un bit* b'*.*

Dans ce jeu, l'adversaire est supposé statique, c'est-à-dire qu'il doit choisir les groupes sur lesquels il veut être défié avant la distribution des clés de déchiffrement aux utilisateurs. Cette restriction est classique, et se retrouve dans tous les modèles utilisés en diffusion : on ne connaît pas de schéma sûr face à un adversaire adaptatif, c'est-à-dire un adversaire qui indique les utilisateurs qu'il souhaite corrompre après la distribution des clés de déchiffrement.

Définition 3.4 (Avantage IND-CPA d'un adversaire) *L'adversaire* $\mathcal{A}$ *gagne le jeu précédent lorsque le bit* b' *qui constitue sa sortie est égal au bit* b *choisi aléatoirement par le challenger.*

L'avantage de l'adversaire $\mathcal{A}$ *dans ce jeu pour les paramètres* $(\lambda, n, (\mathcal{B}(u))_{1 \leqslant u \leqslant n})$ *dépend de sa probabilité de gain, calculée sur l'ensemble des aléas utilisés lors de la génération des clés et du chiffrement, et sur l'aléa utilisé par l'adversaire* $\mathcal{A}$ *:*

$$Adv^{ind}(\lambda, n, (\mathcal{B}(u)), \mathcal{A}) = |2 \Pr[b' = b] - 1|.$$

Définition 3.5 (Sécurité IND-CPA d'un schéma) *Un schéma de diffusion à clé publique avec gestion de groupes est IND-CPA sûr contre des coalitions statiques de toutes tailles si pour tout adversaire* $\mathcal{A}$ *probabiliste polynomial en temps (en* λ*) et pour toute structure d'au plus* l *groupes d'utilisateurs* $(\mathcal{B}(u))_{1 \leqslant u \leqslant n}$*, l'avantage de cet adversaire, noté* $Adv^{ind}(\lambda, n, (\mathcal{B}(u)), \mathcal{A})$*, est une fonction négligeable en* λ*, lorsque* n *et* l *croissent au plus polynomialement en* λ*.*

On peut construire de la même manière des définitions de sécurité plus forte, en permettant à un adversaire d'accéder à des oracles de déchiffrement correspondant aux utilisateurs dont il ne connaît pas les clés de déchiffrement. Cependant, on se limite à un modèle de sécurité plus faible dans la mesure où il existe des transformations génériques (comme celles présentées dans [FO99a, FO99b, OP01]) qui représentent un coût marginal et qui permettent de transformer un schéma sûr dans un modèle faible en un schéma sûr dans un modèle fort (avec oracles de déchiffrement), dans le modèle de l'oracle aléatoire.

3.4 Schéma

La construction proposée dans cette partie est basée sur un problème défini pour des groupes cycliques avec un couplage, comme il en existe de très nombreux exemples depuis l'hypothèse bilinéaire Diffie-Hellman présentée dans [Jou00].

Soit p le nombre premier correspondant au cardinal des groupes cycliques avec couplage utilisés : la longueur de p dépend de la sécurité voulue. Dans ce schéma, on définit, pour chaque groupe d'utilisateurs, un élément public, qu'on appellera abusivement attribut, dans l'ensemble $(\mathbb{Z}/p\mathbb{Z})$. Chaque utilisateur u est décrit par l'ensemble $\Omega(u)$ des attributs qui lui correspondent, étroitement lié à l'ensemble $\mathcal{B}(u)$ défini dans les définitions du modèle (partie 3.3).

La clé de déchiffrement de l'utilisateur u est construite essentiellement à partir de $\Omega(u)$. La taille de cette clé de déchiffrement dépend linéairement de $|\Omega(u)|$, c'est-à-dire du nombre de groupes auxquels appartient l'utilisateur u. Le chiffrement utilise deux ensembles d'attributs, Ω_N et Ω_R (de manière similaire à la définition 3.2 donnée en partie 3.3). L'entête résultant a une taille linéaire en $|\Omega_N| + |\Omega_R|$, nombre de groupes impliqués dans ce chiffrement.

3.4.1 Génération des clés

À partir du paramètre de sécurité λ, la première étape de la génération des clés (***Setup***) consiste à construire des groupes avec un couplage, c'est-à-dire un 6-uplet $(\mathbb{G}_1, \mathbb{G}_2, g_1, g_2, e, p)$ tel que :

- $\mathbb{G}_1$ et $\mathbb{G}_2$ sont deux groupes cycliques d'ordre premier p, notés additivement ;
- g_1 est un générateur de $\mathbb{G}_1$;
- e est un couplage non-dégénéré de $\mathbb{G}_1 \times \mathbb{G}_1$ dans $\mathbb{G}_2$:
  - pour tous $a, b \in (\mathbb{Z}/p\mathbb{Z})$, $e(a\,g_1, b\,g_1) = ab.e(g_1, g_1)$,
  - $g_2 = e(g_1, g_1)$, g_2 est un générateur de $\mathbb{G}_2$;
- la longueur de p est égale à λ ;
- les lois des groupes $\mathbb{G}_1$ et $\mathbb{G}_2$ ainsi que l'application bilinéaire e peuvent être calculées efficacement, c'est-à-dire polynomialement en λ.

On remarque que le couplage est pris dans un contexte symétrique comme par exemple dans [Jou00, BF01] : les deux groupes de départ sont identiques.

Chaque groupe d'utilisateurs $\mathcal{G}_i$ est associé à un attribut μ_i dans $(\mathbb{Z}/p\mathbb{Z})$. Ces attributs doivent être choisis deux à deux différents. Un attribut supplémentaire μ_0 est choisi en plus, dans les mêmes conditions que les attributs précédents : il correspond à un ensemble d'utilisateurs virtuel, qui ne contient aucun utilisateur.

On choisit de plus quatre éléments (α, β, γ et δ) aléatoirement dans $(\mathbb{Z}/p\mathbb{Z})^*$: ils sont choisis uniformément dans $(\mathbb{Z}/p\mathbb{Z})^*$, à l'exception de α qui doit être différent de tous les attributs précédents.

La clé publique de chiffrement est :

$$\mathrm{EK} = \Big(g_1\,,\,\beta\,\gamma\,\delta\,g_1\,,\,(\mu_i)_{0\leqslant i\leqslant l}\,,\,\big(\alpha^i\,g_1\big)_{0\leqslant i\leqslant l}\,,\,\big(\alpha^i\,\gamma\,g_1\big)_{0\leqslant i\leqslant l}\,,\,\big(\alpha^i\,\delta\,g_1\big)_{0\leqslant i\leqslant l}\Big)\,.$$

Dans une version à émetteur unique du schéma, c'est-à-dire où la clé de chiffrement est secrète et seulement divulguée à l'émetteur, on peut utiliser une clé de chiffrement plus petite : $\mathrm{EK} = \big(g_1\,,\,\alpha\,,\,\beta\,,\,\gamma\,,\,\delta\,,\,(\mu_i)_{0\leqslant i\leqslant l}\big)$.

Pour chaque utilisateur $u \in \mathcal{U}$, un élément supplémentaire s_u est choisi aléatoirement et uniformément dans $(\mathbb{Z}/p\mathbb{Z})^*$. Soit $\Omega(u)$ l'ensemble des attributs associés à l'utilisateur u : $\Omega(u) = \{\mu_i \in (\mathbb{Z}/p\mathbb{Z}) \,/\, i \in \mathcal{B}(u)\}$. Soit $l(u)$ la taille de cet ensemble $\Omega(u)$. On définit $\Pi(u) = \prod_{\mu\in\Omega(u)}(\alpha - \mu)$. La clé de déchiffrement de l'utilisateur u est :

$$\mathrm{dk}(u) = \Big(\Omega(u), (\beta + s_u)\,\delta\, g_1\,,\, \gamma\, s_u\, \Pi(u)\, g_1\,,\, \big(\alpha^i\, \gamma\, \delta\, s_u\, g_1\big)_{0\leqslant i<l(u)}\Big)\,.$$

On remarque que la clé de déchiffrement d'un utilisateur dépend d'un paramètre s_u qui n'apparaît pas dans la clé de chiffrement. Il s'agit simplement d'une randomisation des clés de déchiffrement, mise en place pour rendre les clés de déchiffrement de différents utilisateurs non combinables entre elles.

3.4.2 Chiffrement

Conformément à la définition 3.2 de la partie 3.3, le chiffrement (***Encrypt***) utilise la clé publique de chiffrement EK et deux ensembles de groupes d'utilisateurs, $\mathcal{B}_N$ et $\mathcal{B}_R$. On définit tout d'abord les ensembles des attributs associés aux ensembles de groupes[2] : $\Omega_N = \{\mu_i \,/\, i \in \mathcal{B}_N\}$ et $\Omega_R = \{\mu_i \,/\, i \in \mathcal{B}_R\}$, avec $l_N = |\Omega_N|$ et $l_R = |\Omega_R|$.

Si $\Omega_N \cap \Omega_R$ est non-vide, le chiffrement est abandonné : on ne peut demander à un utilisateur d'être simultanément membre et non-membre d'un groupe d'utilisateurs.

Sinon, on choisit aléatoirement et uniformément un élément r dans $(\mathbb{Z}/p\mathbb{Z})^*$. On pose les notations suivantes :

$$\Pi_N = \prod_{\mu\in\Omega_N} (\alpha - \mu)\,, \quad \Pi_R = \prod_{\mu\in\Omega_R} (\alpha - \mu)\,, \quad \Pi_{NR} = \Pi_N\, \Pi_R\,.$$

L'entête et la clé calculés par le chiffrement sont définis par les formules suivantes, où r est un élément choisi aléatoirement et uniformément dans $(\mathbb{Z}/p\mathbb{Z})^*$:

$$\mathrm{hdr} = \Big(\Omega_N, \Omega_R, r\, \Pi_{NR}\, g_1\,,\, \gamma\, r\, \Pi_N\, g_1\,,\, \big(\alpha^i\, \delta\, r\, g_1\big)_{0\leqslant i<l_R}\Big)\,,$$

$$K = \beta\, \gamma\, \delta\, r\, \Pi_N\, g_2.$$

Tous ces éléments peuvent être calculés par un émetteur en utilisant seulement la clé publique de chiffrement EK. En effet, les éléments de l'entête sont calculés par une multiplication par r d'éléments de EK (ou d'une combinaison linéaire de tels éléments). La clé K est calculée par la formule : $K = e\,(\beta\, \gamma\, \delta\, g_1\,,\, r\, \Pi_N\, g_1)$.

[2]Une modification apparaît dans le cas où $\mathcal{B}_R$ est vide. Dans ce cas, Ω_R est défini par : $\Omega_R = \{\mu_0\}$, et $l_R = 1$. Cette modification pratique est utilisée car la sécurité du schéma nécessite que Ω_R soit non-vide : on considère alors que le groupe « virtuel » ne contenant aucun utilisateur est un groupe révoqué.

3.4.3 Déchiffrement

On considère le déchiffrement (***Decrypt***) d'un entête hdr avec la clé de déchiffrement $\mathrm{dk}(u)$ de l'utilisateur u :

$$\begin{cases} \mathrm{dk}(u) = (\Omega(u), \mathrm{dk}_1, \mathrm{dk}_2, \mathrm{dk}_{3,0}, \ldots, \mathrm{dk}_{3,l(u)-1}) \,, \\ \mathrm{hdr} = (\Omega_N, \Omega_R, \mathrm{hdr}_1, \mathrm{hdr}_2, \mathrm{hdr}_{3,0}, \ldots, \mathrm{hdr}_{3,l_R-1}) \,. \end{cases}$$

L'utilisateur u est privilégié pour cet entête si son ensemble d'attributs $\Omega(u)$ contient l'ensemble Ω_N des attributs imposés, et s'il a une intersection vide avec l'ensemble Ω_R des attributs révoqués. Pour obtenir la clé correspondant à l'entête, on commence par utiliser l'algorithme d'Euclide étendu sur les polynômes $\prod_{\mu\in(\Omega_N\cup\Omega_R)}(X-\mu)$ et $\prod_{\mu\in\Omega(u)}(X-\mu)$. Cet algorithme calcule deux polynômes $V(X)$ et $W(X)$, à coefficients dans $(\mathbb{Z}/p\mathbb{Z})$, et de degrés respectivement inférieurs à $l(u)$ et l_R, tels que :

$$V(X) \prod_{\mu\in(\Omega_N\cup\Omega_R)} (X-\mu) + W(X) \prod_{\mu\in\Omega(u)} (X-\mu) = \prod_{\mu\in\Omega_N} (X-\mu).$$

En notant $V(X) = \sum_{0\leqslant i<l(u)} v_i X^i$ et $W(X) = \sum_{0\leqslant i<l_R} w_i X^i$, la formule suivante permet de calculer la clé à partir de ces polynômes :

$$K(\mathrm{dk}(u), \mathrm{hdr}) = e(\mathrm{dk}_1, \mathrm{hdr}_2) \,-\, e\left(\sum_{i=0}^{l(u)-1} v_i \,\mathrm{dk}_{3,i} \,,\, \mathrm{hdr}_1\right) \,-\, e\left(\mathrm{dk}_2 \,,\, \sum_{i=0}^{l_R-1} w_i \,\mathrm{hdr}_{3,i}\right).$$

3.4.4 Consistance

On vérifie désormais la consistance du chiffrement, c'est-à-dire que le déchiffrement d'un entête valide par la clé d'un utilisateur privilégié conduit à la clé associée à l'entête lors du chiffrement.

On considère donc un entête hdr et une clé K issus d'un chiffrement, et une clé de déchiffrement $\mathrm{dk}(u)$ correspondant à un utilisateur u privilégié pour ce chiffrement. Le déchiffrement conduit alors à :

$$K(\mathrm{dk}(u), \mathrm{hdr}) = (\beta + s_u)\ \gamma\,\delta\,r\,\Pi_N\,g_2 \,-\, \gamma\,\delta\,r\,s_u\,V(\alpha)\,\Pi_{NR}\,g_2 \,-\, \gamma\,\delta\,r\,s_u\,W(\alpha)\,\Pi(u)\,g_2.$$

Par définition de V et W, on a la propriété suivante : $V(\alpha)\,\Pi_{NR} + W(\alpha)\Pi(u) = \Pi_N$. La clé calculée lors du déchiffrement est donc exactement égale à la clé K associée à l'entête :

$$K(\mathrm{dk}(u), \mathrm{hdr}) = (\beta + s_u)\ \gamma\,\delta\,r\,\Pi_N\,g_2 \,-\, \gamma\,\delta\,r\,s_u\,\Pi_N\,g_2 = \beta\,\gamma\,\delta\,r\,\Pi_N\,g_2.$$

3.5 Preuve de sécurité

Le schéma défini en partie 3.4 peut être prouvé par des méthodes diverses. La stratégie habituelle consiste à définir une hypothèse de sécurité, à prouver que le schéma

est sûr sous cette hypothèse de sécurité, puis à prouver cette hypothèse dans le modèle générique de groupes avec couplage (voir chapitre 2).

Si on suit cette stratégie, on doit donc définir une nouvelle hypothèse de sécurité, qui consiste en une version décisionnelle du problème GDHE (*General Diffie-Hellman Exponent*) mentionné et prouvé dans la version étendue de [BBG05].

On choisit ici une preuve directement dans le modèle générique de groupes avec couplage. Une approche directe ne présente formellement pas moins de garanties de sécurité qu'en passant par une hypothèse intermédiaire. On commence donc par définir le problème décisionnel correspondant à la sécurité IND-CPA du schéma de diffusion à clé publique avec gestion de groupes proposé précédemment (partie 3.4). On prouve alors la robustesse de ce problème dans le modèle générique des groupes avec couplage.

3.5.1 Problème décisionnel IND-CPA du schéma

- Soient $\mathbb{G}_1$ et $\mathbb{G}_2$ deux groupes cycliques d'ordre premier p, soit e un couplage non dégénéré de $\mathbb{G}_1 \times \mathbb{G}_1$ dans $\mathbb{G}_2$. Soit g_1 un générateur de $\mathbb{G}_1$ et $g_2 = e(g_1, g_1)$.
- Soit $(\mu_i)_{0 \leqslant i \leqslant l}$ une famille d'éléments de $(\mathbb{Z}/p\mathbb{Z})$ deux à deux différents. Pour tout utilisateur $u \in \mathcal{U}$, soit s_u un élément de $(\mathbb{Z}/p\mathbb{Z})^*$, soit $\Omega(u)$ un sous-ensemble de $\{\mu_1, \ldots, \mu_l\}$, de taille $l(u)$.
- Soit α un élément de $(\mathbb{Z}/p\mathbb{Z})^*$ différent des μ_i. Soient β, γ et δ des éléments de $(\mathbb{Z}/p\mathbb{Z})^*$. Soit Ω_N un sous-ensemble de $\{\mu_1, \ldots, \mu_l\}$. Soit Ω_R un sous-ensemble non-vide de $\{\mu_0, \ldots, \mu_l\}$ d'intersection vide avec Ω_N. Soient l_N et l_R les tailles respectives des ensembles Ω_N et Ω_R.
- Soit b un bit, et K_{1-b} un élément de ${\mathbb{G}_2}^*$.

La clé publique de chiffrement est :

$$\mathrm{EK} = \Big(g_1 \,,\, \beta\,\gamma\,\delta\, g_1 \,,\, (\mu_i)_{0 \leqslant i \leqslant l} \,,\, \big(\alpha^i\, g_1\big)_{0 \leqslant i \leqslant l} \,,\, \big(\alpha^i\, \gamma\, g_1\big)_{0 \leqslant i \leqslant l} \,,\, \big(\alpha^i\, \delta\, g_1\big)_{0 \leqslant i \leqslant l} \Big).$$

Pour tout utilisateur $u \in \mathcal{U}$, on définit $\Pi(u) = \prod_{\mu \in \Omega(u)} (\alpha - \mu)$. La clé de déchiffrement $\mathrm{dk}(u)$ de l'utilisateur u est :

$$\mathrm{dk}(u) = \Big(\Omega(u), (\beta + s_u)\,\delta\, g_1 \,,\, \gamma\, s_u\, \Pi(u)\, g_1 \,,\, \big(\alpha^i\, \gamma\, \delta\, s_u\, g_1\big)_{0 \leqslant i < l(u)} \Big).$$

On définit : $\Pi_N = \prod_{\mu \in \Omega_N} (\alpha - \mu)$, $\Pi_R = \prod_{\mu \in \Omega_R} (\alpha - \mu)$ et $\Pi_{NR} = \Pi_N\, \Pi_R$. L'entête hdr et la clé K_b sont :

$$\mathrm{hdr} = \Big(\Omega_N, \Omega_R, r\, \Pi_{NR}\, g_1 \,,\, \gamma\, r\, \Pi_N\, g_1 \,,\, \big(\alpha^i\, \delta\, r\, g_1\big)_{0 \leqslant i < l_R} \Big),$$

$$K_b = \beta\, \gamma\, \delta\, r\, \Pi_N\, g_2.$$

Le problème décisionnel consiste à deviner b en ayant connaissance de la clé publique EK, de l'entête hdr, des clés K_0 et K_1, et de l'ensemble des clés de déchiffrement des utilisateurs révoqués, c'est-à-dire des utilisateurs appartenant à l'ensemble $\mathcal{R} = \{u \in \mathcal{U} \,/\, \Omega(u) \cap \Omega_N \neq \Omega_N \text{ ou } \Omega(u) \cap \Omega_R \neq \emptyset\}$.

3.5.2 Interprétation dans le modèle générique

On s'appuie ici sur la présentation du modèle générique des groupes avec couplage réalisée dans le chapitre 2. La première partie de la preuve consiste à montrer qu'il n'existe pas de « formule universelle » permettant de calculer la clé K_b à partir de la clé de chiffrement, de l'entête, et des clés de déchiffrement des utilisateurs révoqués. La seconde partie prouve qu'un adversaire ne peut pas distinguer la clé K_b d'un élément aléatoire dans le modèle générique des groupes avec couplage, sachant qu'il n'existe pas de formule de calcul.

3.5.2.1 Absence de formule universelle

On considère l'anneau $\mathcal{P}$ des polynômes à coefficients dans $(\mathbb{Z}/p\mathbb{Z})$ en les variables $A, B, C, D, R, \{S_u, u \in \mathcal{R}\}$. Chacune de ces variables correspond à un élément choisi aléatoirement dans le problème décisionnel, mais qui n'est pas révélé explicitement : A pour α, B pour β, C pour γ, D pour δ, R pour r et S_u pour s_u (pour chaque $u \in \mathcal{U}$). Tous les éléments donnés à un attaquant sur ce problème peuvent s'interpréter comme des polynômes en ces variables, c'est-à-dire en des éléments de $\mathcal{P}$. C'est également le cas de la clé K_b correspondant à l'entête.

L'objet de cette partie de la preuve est de montrer que le polynôme correspondant à la clé K_b ne s'écrit pas comme une combinaison linéaire de produits de polynômes associés aux éléments connus de l'adversaire. De cette manière, on prouve qu'il n'existe pas de formule générique permettant de calculer la clé K_b à partir des éléments donnés à l'adversaire, en utilisant seulement les lois des groupes et le couplage.

Soit $\mathcal{D}$ l'ensemble des polynômes correspondants aux éléments donnés à l'adversaire dans le problème décisionnel. Cet ensemble contient 1, $B\,C\,D$, $R\,\Pi_{NR}(A)$, $C\,R\,\Pi_N(A)$ et les polynômes suivants :

- A^i , $A^i\,C$ et $A^i\,D$ pour tout $i \in \{0, \ldots, l\}$,
- $(B + S_u)\,D$ et $C\,S_u\,\Pi_u(A)$, pour tout $u \in \mathcal{R}$,
- $A^i\,C\,D\,S_u$, pour tout $u \in \mathcal{R}$ et $i \in \{0, \ldots, l(u) - 1\}$,
- $A^i\,D\,R$ pour tout $i \in \{0, \ldots, l_R - 1\}$,

où

$$\Pi_N(X) = \prod_{\mu \in \Omega_N} (X - \mu), \quad \Pi_R(X) = \prod_{\mu \in \Omega_R} (X - \mu),$$

$$\Pi_u(X) = \prod_{\mu \in \Omega(u)} (X - \mu), \quad \Pi_{NR}(X) = \Pi_N(X)\,\Pi_R(X).$$

Lemme 3.1 (Absence de formule universelle) *Soit $\mathcal{M}$ le sous-$\mathbb{Z}$-module de $\mathcal{P}$ engendré par les produits de deux éléments de $\mathcal{D}$. Lorsque $l_R \leqslant \sqrt[4]{p}/4$ et que pour tout u, $l(u) \leqslant \sqrt[4]{p}/4$, le polynôme $B\,C\,D\,R\,\Pi_N(A)$ appartient à ce module $\mathcal{M}$ avec une probabilité inférieure ou égale à $1/\sqrt{p}$, cette probabilité étant prise sur l'ensemble de tous les choix possibles pour les attributs μ_i des groupes dans $(\mathbb{Z}/p\mathbb{Z})$.*

Preuve On considère l'ensemble $\mathcal{D}'$ des produits de deux éléments de $\mathcal{D}$. Par définition, cet ensemble $\mathcal{D}'$ engendre le module $\mathcal{M}$.

On suppose désormais que le polynôme $B\,C\,D\,R\,\Pi_N(A)$ appartient à $\mathcal{M}$. Il est donc une combinaison $\mathbb{Z}$-linéaire d'éléments de $\mathcal{D}'$. En considérant les éléments de $\mathcal{D}'$ comme des polynômes univariés en la variable C, on remarque que ces polynômes sont homogènes de degré 0, 1 ou 2. Le polynôme $B\,C\,D\,R\,\Pi_N(A)$ peut donc s'écrire comme une combinaison $\mathbb{Z}$-linéaire des seuls polynômes de $\mathcal{D}'$ linéaires en C. Le même raisonnement s'applique également aux variables D et R.

On considère désormais les polynômes de $\mathcal{D}'$ comme multivariés en les variables $B, \{S_u, u \in \mathcal{R}\}$. Ils sont également homogènes de degré 0, 1 ou 2 : de la même manière, le polynôme $B\,C\,D\,R\,\Pi_N(A)$ doit pouvoir s'écrire comme une combinaison de termes de degré 1 en ces variables.

On s'intéresse donc spécifiquement aux polynômes de $\mathcal{D}'$ qui sont simultanément linéaires en C, en D et en R, tout en étant globalement de degré 1 en les variables B, $\{S_u, u \in \mathcal{R}\}$. Ces polynômes sont listés dans les ensembles suivants :

$$\mathcal{D}'_1 = \{B\,C\,D\,R\,\Pi_{NR}(A)\}\,,$$

$$\mathcal{D}'_2 = \{(B + S_u)\,C\,D\,R\,\Pi_N(A) \,/\, u \in \mathcal{R}\}\,,$$

$$\mathcal{D}'_3 = \left\{A^i\,C\,D\,R\,S_u\,\Pi_u(A) \,/\, u \in \mathcal{R}, i \in \{0, \ldots, l_R - 1\}\right\},$$

$$\mathcal{D}'_4 = \left\{A^i\,C\,D\,R\,S_u\,\Pi_{NR}(A) \,/\, u \in \mathcal{R}, i \in \{0, \ldots, l(u) - 1\}\right\}.$$

Le polynôme dans $\mathcal{D}'_1$ n'est pas le polynôme $B\,C\,D\,R\,\Pi_N(A)$, puisque l'ensemble Ω_R est non-vide. Comme B n'apparaît que dans les polynômes des ensembles $\mathcal{D}'_1$ et $\mathcal{D}'_2$, au moins l'un des polynômes de $\mathcal{D}'_2$ doit être utilisé dans la combinaison linéaire.

Dans cette combinaison linéaire, on doit donc annuler au moins un des termes de la forme $S_u\,C\,D\,R\,\Pi_N(A)$, puisque ces termes sont issus des polynômes dans $\mathcal{D}'_2$ et qu'ils sont linéairement indépendants. En ne considérant que les termes linéaires en la variable S_u (où u est fixé) dans les ensembles $\mathcal{D}'_3$ et $\mathcal{D}'_4$, il est nécessaire de construire une relation de la forme :

$$\Pi_N(A) = \Big(\sum_{i=0}^{l_R-1} \lambda_i\,A^i\Big)\,\Pi_u(A) + \Big(\sum_{i=0}^{l(u)-1} \lambda'_i\,A^i\Big)\,\Pi_{NR}(A). \tag{3.1}$$

L'utilisateur u est par hypothèse révoqué : soit il appartient à un groupe révoqué, soit il n'appartient pas à un groupe imposé. Ces deux cas sont traités séparément :

– Si $\Omega(u) \cap \Omega_R \neq \emptyset$ (appartenance à un groupe révoqué), on considère un attribut μ dans cette intersection. Le polynôme $(A - \mu)$ divise les polynômes $\Pi_u(A)$ et $\Pi_{NR}(A)$, et par conséquent la partie droite de l'équation (3.1). Par hypothèse, Ω_N et Ω_R ont une intersection vide : $(A - \mu)$ ne peut donc pas diviser $\Pi_N(A)$ et la relation (3.1) est donc impossible.

– Dans le cas contraire, Ω_N est nécessairement non-contenu dans $\Omega(u)$ (non-appartenance à un groupe imposé), c'est-à-dire que $\Pi_N(A)$ ne divise pas $\Pi_u(A)$. Soit $\pi_u(A)$ le polynôme dont les racines sont les attributs « inutiles » : ses racines sont les attributs non-requis par le chiffré, c'est-à-dire dans $\Omega(u)$ mais pas dans Ω_N. On a $\pi_u(A) = \Pi_u(A) \,/\, \mathrm{pgcd}(\Pi_u(A), \Pi_N(A))$.
Comme $\Pi_N(A)$ divise $\Pi_{NR}(A)$, l'équation (3.1) impose que $\Pi_N(A)$ divise le produit $(\sum_{i=0}^{l_R-1} \lambda_i A^i)\, \Pi_u(A)$. Comme $\pi_u(A)$ divise $\Pi_u(A)$, et comme $\pi_u(A)$ est premier avec $\Pi_N(A)$, le polynôme $\Pi_N(A)$ divise $(\sum_{i=0}^{l_R-1} \lambda_i A^i)\, \Pi_u(A)/\pi_u(A)$. On définit alors le polynôme $Q(A)$ par la formule :

$$Q(A) = \frac{(\sum_{i=0}^{l_R-1} \lambda_i\, A^i)\ \Pi_u(A)}{\pi_u(A)\ \Pi_N(A)}.$$

Par définition de $\pi_u(A)$, $\Pi_u(A)$ divise $\pi_u(A)\ \Pi_N(A)$. D'autre part, par hypothèse, il n'y a pas égalité (il existe une racine de $\Pi_N(A)$ qui n'est pas racine de $\Pi_u(A)$). On en déduit que le polynôme $Q(A)$ divise strictement $\sum_{i=0}^{l_R-1} \lambda_i A^i$. Le degré de $Q(A)$ est donc au plus $l_R - 2$, et le polynôme $Q(A)$ vérifie :

$$\Big(\sum_{i=0}^{l_R-1} \lambda_i A^i\Big)\, \Pi_u(A) = \Pi_N(A)\ Q(A)\ \pi_u(A).$$

Ainsi, l'équation (3.1) est équivalente à la relation suivante :

$$Q(A)\, \pi_u(A) + \Big(\sum_{i=0}^{l(u)-1} \lambda'_i\, A^i\Big)\, \Pi_R(A) = 1, \quad \text{avec } \deg(Q) < \deg(\Pi_R) - 1.$$

Les polynômes $\pi_u(A)$ et $\Pi_R(A)$ sont scindés à racines simples, et leurs racines sont tirées aléatoirement. Le lemme 3.3 prouve alors qu'une telle relation existe avec une probabilité au plus égale à $1/\sqrt{p}$ puisque les degrés des polynômes vérifient la borne demandée.

Dans un cas, la relation 3.1 ne peut exister, dans l'autre, elle ne peut exister qu'avec une probabilité au plus égale à $1/\sqrt{p}$. Avec probabilité supérieure à $1-1/\sqrt{p}$, il y a donc contradiction avec l'hypothèse de départ selon laquelle le polynôme $B\,C\,D\,R\,\Pi_N(A)$ appartenait à $\mathcal{M}$, ce qui prouve le lemme. □

Il reste juste à prouver le résultat utilisé dans le second cas. On procède en deux étapes : dans le lemme 3.2, on calcule une probabilité s'appuyant sur des polynômes unitaires premiers entre eux. Ce résultat étant insuffisant, on travaille spécifiquement sur les polynômes scindés à racines simples dans le lemme 3.3.

On considère deux polynômes univariés en une variable X définis sur $(\mathbb{Z}/p\mathbb{Z})$ et unitaires : P_1 de degré d_1 et P_2 de degré d_2. On suppose que ces deux polynômes sont premiers entre eux. Par le théorème de Bezout, il existe deux polynômes U_1 et U_2 tels que :

$$U_1 P_1 - U_2 P_2 = 1, \quad \text{avec } \deg(U_1) < d_2 \text{ et } \deg(U_2) < d_1. \tag{3.2}$$

La condition sur les degrés garantit de plus l'unicité du couple (U_1, U_2) vérifiant l'équation (3.2). L'algorithme d'Euclide permet un calcul effectif de ce couple de polynômes. Le lemme 3.2 s'intéresse à la probabilité que $\deg U_1 < d_2 - 1$.

Lemme 3.2 (Bezout dégénéré) *Pour tout $(d_1, d_2) \in \mathbb{N} \times \mathbb{N}^*$, pour tout p premier supérieur à $(d_1 + d_2)^2$, la probabilité, définie sur l'ensemble des couples (P_1, P_2) de polynômes unitaires de $(\mathbb{Z}/p\mathbb{Z})[X]$ premiers entre eux et de degrés (d_1, d_2), que le couple (U_1, U_2) de polynômes défini de manière unique par la relation (3.2) vérifie $\deg U_1 < d_2 - 1$ est majorée par $1/\sqrt{p}$.*

Preuve Soient P_1 et P_2 deux polynômes unitaires de $(\mathbb{Z}/p\mathbb{Z})[X]$, de degrés $d_1 \in \mathbb{N}$ et $d_2 \in \mathbb{N}^*$. Ces polynômes sont non-premiers entre eux s'ils vérifient la relation de Sylvester, c'est-à-dire qu'il existe deux polynômes unitaires (donc non-nuls) V_1 et V_2 tels que :

$$V_1 P_1 = V_2 P_2, \quad \text{avec } \deg(V_1) < d_2 \text{ et } \deg(V_2) < d_1. \tag{3.3}$$

Cette relation est équivalente au fait que la famille suivante soit liée dans $(\mathbb{Z}/p\mathbb{Z})[X]$, vu comme un $(\mathbb{Z}/p\mathbb{Z})$-espace vectoriel :

$$\left(\{P_1(X)\, X^k \,/\, k \in \{0, \ldots, d_2 - 1\}\}, \{P_2(X)\, X^k \,/\, k \in \{0, \ldots, d_1 - 1\}\} \right).$$

On définit les coefficients des polynômes P_1 et P_2 par les formules suivantes : $P_1(X) = X^{d_1} + \sum_{i=0}^{d_1-1} \nu_i X^i$ et $P_2(X) = X^{d_2} + \sum_{i=0}^{d_2-1} \nu'_i X^i$. On exprime le fait que la famille soit liée par l'annulation d'un déterminant de Sylvester de dimension $d_1 + d_2$:

$$\det \begin{pmatrix} \nu_0 & 0 & \cdots & \cdots & 0 & \nu'_0 & 0 & \cdots & 0 \\ \nu_1 & \nu_0 & \ddots & & \vdots & \nu'_1 & \nu'_0 & \ddots & \vdots \\ \vdots & \nu_1 & \ddots & \ddots & \vdots & \vdots & \nu'_1 & \ddots & 0 \\ \nu_{d_1-1} & \vdots & \ddots & \ddots & 0 & \vdots & \vdots & \ddots & \nu'_0 \\ 1 & \nu_{d_1-1} & & \ddots & \nu_0 & \nu'_{d_2-1} & \vdots & & \nu'_1 \\ 0 & 1 & \ddots & & \nu_1 & 1 & \nu'_{d_2-1} & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots & 0 & 1 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \nu_{d_1-1} & \vdots & \ddots & \ddots & \nu'_{d_2-1} \\ 0 & \cdots & \cdots & 0 & 1 & 0 & \cdots & 0 & 1 \end{pmatrix} = 0.$$

En développant ce déterminant, on obtient un polynôme de degré $d_1 + d_2 - 1$ en les $d_1 + d_2$ coefficients de P_1 et P_2, sur $(\mathbb{Z}/p\mathbb{Z})$. Le lemme 1 donné dans [Sch80] prouve que la probabilité sur l'ensemble des valeurs des coefficients dans $(\mathbb{Z}/p\mathbb{Z})$ que ce polynôme s'annule est majorée par $(d_1 + d_2 - 1)/p$. Il y a donc au moins $(p + 1 - d_1 - d_2)\, p^{d_1+d_2-1}$ couples de polynômes unitaires de degrés d_1 et d_2 premiers entre eux.

On suppose désormais que les polynômes unitaires P_1 et P_2 sont premiers entre eux, et que le couple de polynômes (U_1, U_2) défini par la relation (3.2) vérifie : $\deg U_1 < d_2 - 1$.

On en déduit immédiatement que $\deg U_2 < d_1 - 1$. L'interprétation de la relation (3.2) avec ces conditions de degré, dans le $(\mathbb{Z}/p\mathbb{Z})$-espace vectoriel $(\mathbb{Z}/p\mathbb{Z})[X]$, implique que la famille suivante est liée (il n'y a cependant pas équivalence) :

$$\Big(1, \{P_1(X)\,X^k \,/\, k \in \{0,\ldots,d_2-2\}\}, \{P_2(X)\,X^k \,/\, k \in \{0,\ldots,d_1-2\}\}\Big).$$

De la même manière que précédemment, on transforme cette relation en une annulation d'un déterminant de dimension d_1+d_2-1 impliquant les coefficients des polynômes P_1 et P_2 :

$$\det \begin{pmatrix}
1 & \nu_0 & 0 & \cdots & \cdots & 0 & \nu'_0 & 0 & \cdots & 0 \\
0 & \nu_1 & \nu_0 & \ddots & & \vdots & \nu'_1 & \nu'_0 & \ddots & \vdots \\
\vdots & \vdots & \nu_1 & \ddots & \ddots & \vdots & \vdots & \nu'_1 & \ddots & 0 \\
\vdots & \nu_{d_1-1} & \vdots & \ddots & \ddots & 0 & \vdots & \vdots & \ddots & \nu'_0 \\
\vdots & 1 & \nu_{d_1-1} & & \ddots & \nu_0 & \nu'_{d_2-1} & \vdots & & \nu'_1 \\
\vdots & 0 & 1 & \ddots & & \nu_1 & 1 & \nu'_{d_2-1} & & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & \vdots & 0 & 1 & \ddots & \vdots \\
\vdots & \vdots & & \ddots & \ddots & \nu_{d_1-1} & \vdots & \ddots & \ddots & \nu'_{d_2-1} \\
0 & 0 & \cdots & \cdots & 0 & 1 & 0 & \cdots & 0 & 1
\end{pmatrix} = 0.$$

En développant ce déterminant, on obtient un polynôme de degré $d_1 + d_2 - 3$ en les $d_1 + d_2$ coefficients de P_1 et P_2, sur $(\mathbb{Z}/p\mathbb{Z})$. Une nouvelle utilisation du lemme 1 donné dans [Sch80] prouve que la probabilité sur l'ensemble des valeurs des coefficients dans $(\mathbb{Z}/p\mathbb{Z})$ que ce polynôme s'annule est majorée par $(d_1 + d_2 - 3)/p$. Il y a donc au plus $(d_1 + d_2 - 3)\,p^{d_1+d_2-1}$ couples de polynômes unitaires de degrés d_1 et d_2 premiers entre eux, et tels que l'équation de Bezout fasse intervenir un polynôme U_1 de degré strictement inférieur à $d_2 - 1$.

Il suffit dès lors de calculer le quotient des tailles des deux ensembles calculés pour obtenir une majoration de la probabilité pour un couple de polynômes unitaires premiers entre eux de vérifier l'équation de Bezout (3.2) avec $\deg(U_1) < d_2 - 1$. Lorsque $d_1 + d_2$ est inférieur à $\sqrt{p}$, cette probabilité est majorée par :

$$\frac{d_1 + d_2 - 3}{p + 1 - d_1 - d_2} \leqslant \frac{1}{\sqrt{p}}.$$ □

On reprend désormais cette technique de preuve pour limiter l'étude aux seuls polynômes scindés à racines simples.

Lemme 3.3 (Bezout dégénéré et polynômes scindés) *Pour tout $(d_1, d_2) \in \mathbb{N} \times \mathbb{N}^*$, pour tout p premier supérieur à $2^8 \max(d_1, d_2)^4$, la probabilité, définie sur l'ensemble des couples (P_1, P_2) de polynômes unitaires de $(\mathbb{Z}/p\mathbb{Z})[X]$, scindés, à racines simples, premiers entre eux et de degrés (d_1, d_2), que le couple (U_1, U_2) de polynômes défini de manière unique par la relation (3.2) vérifie $\deg U_1 < d_2 - 1$ est majorée par $1/\sqrt{p}$.*

Preuve Pour prouver ce lemme, on utilise la même méthode que dans le lemme 3.2. On dénombre d'abord le nombre de couples de polynômes unitaires, scindés, à racines simples, premiers entre eux et de degrés respectifs d_1 et d_2. Ensuite, on donne un majorant au nombre de ces polynômes vérifiant une relation de Bezout dégénérée. Cette seconde étape peut être réalisée comme dans le lemme précédent, en remarquant que les coefficients d'un polynôme P scindé de degré d peuvent s'écrire eux-mêmes comme des polynômes de degré au plus d en les racines du polynôme P (il s'agit des polynômes symétriques).

Le nombre de couples de polynômes unitaires, scindés, à racines simples et de degrés respectifs d_1 et d_2 est simplement : $C_p^{d_1}\, C_p^{d_2}$. Le nombre de couples pour lesquels les polynômes ne sont pas premiers entre eux se calcule également par un dénombrement : $p\, C_{p-1}^{d_1-1}\, C_{p-1}^{d_2-1}$. On a donc $C_p^{d_1}\, C_p^{d_2}\, (1 - d_1 d_2/p)$ couples (P_1, P_2) de polynômes unitaires de $(\mathbb{Z}/p\mathbb{Z})[X]$, scindés, à racines simples, premiers entre eux et de degrés (d_1, d_2).

On considère désormais le second déterminant du lemme précédent, qui est nécessairement nul lorsqu'une relation de Bezout dégénérée apparaît. En remplaçant chaque coefficient d'un polynôme par un polynôme symétrique de degré au plus $\max(d_1, d_2)$ en les racines de P_1 et de P_2, et en développant, on obtient un polynôme de degré au plus $\max(d_1, d_2)\,(d_1 + d_2 - 3)$ en les racines des polynômes P_1 et P_2. Il y a donc au plus $\max(d_1, d_2)\,(d_1 + d_2 - 3)\, p^{d_1+d_2-1}$ vecteurs de racines de P_1 et de P_2 annulant le déterminant. Pour chaque vecteur de racines de P_1 et de P_2 annulant le déterminant, toute permutation des racines de P_1 et toute permutation des racines de P_2 laissent les polynômes P_1 et P_2 invariants. Comme seuls les polynômes scindés à racines simples sont considérés, on peut diviser le nombre de vecteurs de racines de P_1 et de P_2 par $d_1!\, d_2!$ pour obtenir un majorant du nombre de polynômes unitaires scindés à racines simples de degrés respectifs d_1 et d_2 vérifiant une relation de Bezout dégénérée :

$$\frac{\max(d_1, d_2)\,(d_1 + d_2 - 3)\, p^{d_1+d_2-1}}{d_1!\, d_2!}.$$

La probabilité, définie sur l'ensemble des couples (P_1, P_2) de polynômes unitaires de $(\mathbb{Z}/p\mathbb{Z})[X]$, scindés, à racines simples, premiers entre eux et de degrés (d_1, d_2), que le couple (U_1, U_2) de polynômes défini de manière unique par la relation (3.2) vérifie $\deg U_1 < d_2 - 1$ est majorée par :

$$\frac{\max(d_1, d_2)\,(d_1 + d_2 - 3)\, p^{d_1+d_2-1}}{d_1!\, d_2!\, C_p^{d_1}\, C_p^{d_2}\, \left(1 - \frac{d_1\, d_2}{p}\right)} \;\leqslant\; \frac{2\, \max(d_1, d_2)^2}{p\left(1 - \frac{d_1}{p}\right)^{d_1} \left(1 - \frac{d_2}{p}\right)^{d_2} \left(1 - \frac{d_1\, d_2}{p}\right)}.$$

Lorsque d_1 et d_2 sont majorés par $\sqrt{p}/2$, on obtient la borne supérieure suivante :

$$\frac{16\, \max(d_1, d_2)^2}{p}.$$

Lorsque de plus $p \geqslant 2^8 \max(d_1, d_2)^4$, cette dernière borne est inférieure ou égale à $1/\sqrt{p}$, ce qui clôt la démonstration du lemme. □

3.5.2.2 Indistinguabilité dans le modèle générique

La section précédente prouve que dans le modèle générique, l'adversaire ne peut obtenir la clé en résultat direct d'opérations de groupes et de couplages qu'avec une probabilité négligeable. Sa seule possibilité de gain dans le jeu IND-CPA repose donc sur sa capacité à générer des collisions dans le modèle générique : on borne les probabilités correspondant à ces collisions de la même manière que dans les sections 2.4.3 et 2.4.4, c'est-à-dire en utilisant le lemme 2.1.

Toutes les notations utilisées désormais proviennent de la section 2.4.1 du chapitre précédent. Elles sont nécessaires pour prouver le corollaire suivant du lemme 2.1.

Corollaire 3.1 *Soit* $k < (\sqrt{p/(3l+6)} - n\,l - 2n - 4l - 8)/3$. *On suppose que le polynôme associé à la clé ne peut être obtenu par combinaison linéaire de produits de polynômes associés aux données du problème IND-CPA décrit en section 3.5.2. On considère des suites de listes* R *et* S *basées sur ce même problème. La probabilité qu'il y ait une collision après* k *appels aux oracles, c'est-à-dire dans les listes* $(\mathsf{R}_k, \mathsf{S}_k)$*, est majorée par :*

$$\frac{6\,k\,(l+2)\,(n\,l+2n+4l+3k+8)}{p-3\,(l+2)\,(n\,l+2n+4l+3k+8)^2}.$$

Preuve Dans notre contexte, l'ensemble R_0 contient au plus $nl+2n+4l+7$ éléments. De plus ses polynômes sont de degrés majorés par $l+2$. L'ensemble S_0 contient 4 éléments : l'élément neutre, le générateur, la clé fictive (son polynôme est une nouvelle inconnue) et la clé réelle (son polynôme est de degré majoré par $l+4$).

Lors des appels aux oracles, des polynômes de degré au plus $2l+4$ sont insérés dans les listes. On a donc $d = 2l+4$. On a $\rho_0 + \sigma_0 \leqslant nl + 2n + 4l + 11$, on en déduit classiquement que pour tout $i \in \{0, \ldots, k-1\}$, $\rho_i + \sigma_i \leqslant nl + 2n + 4l + 11 + 3i$. Avec la condition sur k imposée, on vérifie bien la propriété que pour tout $i \in \{0, \ldots, k-1\}$, $\rho_i + \sigma_i < \sqrt{2p/3d}$.

Pour tout $i \in \{0, \ldots, k-1\}$, on applique le lemme 2.1. La probabilité de collision après $i+1$ appels aux oracles sachant qu'il n'y a pas eu collision après i appels aux oracles est majorée par :

$$\frac{6d\,(\rho_i + \sigma_i)}{2p - 3d\,(\rho_i + \sigma_i)^2}.$$

La probabilité globale de collision après k appels aux oracles est donc majorée par :

$$\begin{aligned}
1 - \prod_{i=0}^{k-1} \left(1 - \frac{6d}{2p - 3d\,(\rho_i + \sigma_i)^2}\right) &\leqslant 1 - \left(1 - \frac{6d\,(\rho_{k-1} + \sigma_{k-1})}{2p - 3d\,(\rho_{k-1} + \sigma_{k-1})^2}\right)^k \\
&\leqslant \frac{6\,k\,d\,(\rho_{k-1} + \sigma_{k-1})}{2p - 3d\,(\rho_{k-1} + \sigma_{k-1})^2} \\
&\leqslant \frac{6\,k\,(l+2)\,(n\,l+2n+4l+3k+8)}{p-3\,(l+2)\,(n\,l+2n+4l+3k+8)^2}.
\end{aligned}$$

□

3.5.2.3 Avantage global de l'adversaire

Les deux propriétés précédentes (lemme 3.1 et corollaire 3.1) donnent le résultat global suivant :

Théorème 3.1 *Soit* $k < (\sqrt{p/(3l+6)} - n\,l - 2n - 4l - 8)/3$. *On suppose que* $l_R \leqslant \sqrt[4]{p}/4$ *et que pour tout* u, $l(u) \leqslant \sqrt[4]{p}/4$. *Dans le modèle générique des groupes avec couplage, l'avantage sur le problème IND-CPA du schéma de diffusion présenté en 3.4 d'un adversaire réalisant au plus* k *requêtes aux oracles est borné par :*

$$\frac{6\,k\,(l+2)\,(n\,l+2n+4l+3k+8)}{(1-1/\sqrt{p})\left(p-3\,(l+2)\,(n\,l+2n+4l+3k+8)^2\right)}.$$

Preuve On divise simplement la probabilité maximale obtenue suite à l'application du corollaire 3.1 par le facteur $1-1/\sqrt{p}$ qui minore la probabilité que le polynôme associé à la clé n'appartienne pas au sous-$\mathbb{Z}$-module engendré par les produits d'éléments de $\mathcal{D}$, en vertu du lemme 3.1.

On obtient ainsi la probabilité qu'il n'y ait pas collision dans les listes $(\mathsf{R}_k, \mathsf{S}_k)$. Or en l'absence de collisions, dans un problème décisionnel, l'adversaire ne peut que tirer aléatoirement son bit b' : les données correspondant au cas $b = 0$ et $b = 1$ sont indistinguables au vu des résultats des appels aux oracles. On en déduit la borne de son avantage. □

On considère désormais que le paramètre de sécurité augmente : les valeurs n, l et k sont polynomiales en le paramètre de sécurité λ, alors que p est exponentiel en ce même paramètre. Les conditions d'utilisation du théorème, c'est-à-dire les inégalités requises sur p, l et n issues du lemme 3.1 et du corollaire 3.1, sont asymptotiquement vérifiées. La borne donnée à l'avantage de l'adversaire est une fonction négligeable du paramètre de sécurité, ce qui conclut la preuve de sécurité du schéma de diffusion dans le modèle générique de groupes avec couplage.

3.5.3 Variante pour des adversaires adaptatifs

La preuve de sécurité du schéma présenté précédemment s'appuie sur un calcul de probabilités basé sur les valeurs des attributs. Or ces attributs sont fixés lors de la génération des clés, et ils sont connus de l'adversaire. Si la preuve est convaincante dans le modèle de sécurité présenté en 3.3.4, elle n'est pas totalement satisfaisante d'un point de vue concret : un adversaire adaptatif, c'est-à-dire en mesure de choisir les ensembles d'utilisateurs sélectionnés et révoqués pour le chiffré qui lui sera soumis après la génération des clés, peut dans certaines circonstances avoir à coup sûr la capacité de déchiffrer ce message chiffré, et donc de gagner le jeu. Le schéma est donc vulnérable face à des adversaires adaptatifs, et pas seulement non prouvé dans ce cadre.

On présente donc une variante du schéma initial, pour lequel ce phénomène n'est plus présent. Plus précisément, cette variante peut être prouvée pour un adversaire adaptatif, dans le modèle générique des groupes avec couplage.

3.5.3.1 Variante du schéma

La modification du schéma porte sur le mécanisme de chiffrement. Ainsi, la génération des clés et le déchiffrement restent inchangés par rapport au schéma initial. Lors d'un chiffrement, un attribut additionnel ω est choisi aléatoirement dans $(\mathbb{Z}/p\mathbb{Z}) \setminus \{\mu_1, \ldots, \mu_l\}$: cet attribut est alors traité comme un attribut révoqué pour le chiffré.

Dans cette variante, l'utilisation de l'attribut μ_0 est inutile. Pour rappel, cet attribut fictif, correspondant à l'ensemble vide, devait être utilisé dans le schéma initial lorsqu'aucun attribut n'était révoqué dans un chiffrement. Dans la variante, l'attribut choisi aléatoirement ω est révoqué, et l'ensemble des attributs révoqués n'est donc jamais vide.

3.5.3.2 Preuve de la variante

La preuve de la variante suit exactement le même cheminement que celle du schéma original. La modification intervient seulement dans le calcul de la probabilité d'occurrence d'une relation de Bezout dégénérée, qui était traité dans le lemme 3.3. Avec l'utilisation d'un attribut choisi aléatoirement au moment du chiffrement, on peut utiliser directement le lemme suivant :

Lemme 3.4 (Bezout dégénéré et polynômes randomisés) *Pour tout $(d_1, d_2) \in \mathbb{N} \times \mathbb{N}^*$, pour tout p premier, pour tout couple de polynômes unitaires (P_1, P_2) à coefficients dans $(\mathbb{Z}/p\mathbb{Z})$, de degrés (d_1, d_2), et premiers entre eux, il existe au plus $(d_2 - 1)$ éléments $\omega \in (\mathbb{Z}/p\mathbb{Z})$ pour lesquels il existe un couple de polynômes (U_1, U_2) vérifiant :*

$$U_1(A)\,(A-\omega)\,P_1(A) - U_2(A)\,P_2(A) = 1\,, \quad \textit{avec} \quad \deg U_1 < d_2 - 1.$$

Preuve Soit p premier, soit $(d_1, d_2) \in \mathbb{N}\times\mathbb{N}^*$, soient P_1 et P_2 deux polynômes unitaires à coefficients dans $(\mathbb{Z}/p\mathbb{Z})$, premiers entre eux et de degrés respectifs d_1 et d_2. On suppose qu'il existe deux éléments distincts ω et $\tilde{\omega}$ de $(\mathbb{Z}/p\mathbb{Z})$ et quatre polynômes U_1, U_2, $\tilde{U}_1$ et $\tilde{U}_2$ vérifiant :

$$\begin{cases} U_1(A)\,(A-\omega)\,P_1(A) - U_2(A)\,P_2(A) = 1, & \deg U_1 < d_2 - 1\,, \quad \deg U_2 < d_1\,, \\ \tilde{U}_1(A)\,(A-\tilde{\omega})\,P_1(A) - \tilde{U}_2(A)\,P_2(A) = 1, & \deg \tilde{U}_1 < d_2 - 1\,, \quad \deg \tilde{U}_2 < d_1\,. \end{cases}$$

En calculant la différence des deux équations, on obtient $V_1\,P_1 = V_2\,P_2$, avec :

$$\begin{cases} V_1(A) \;=\; \big(U_1(A) - \tilde{U}_1(A)\big)A + \tilde{\omega}\,\tilde{U}_1(A) - \omega\,U_1(A), & \deg V_1 < d_2\,, \\ V_2(A) \;=\; U_2(A) - \tilde{U}_2(A), & \deg V_2 < d_1\,. \end{cases}$$

Si $V_2 \neq 0$, les polynômes P_1 et P_2 étant premiers entre eux, la relation $V_1\,P_1 = V_2\,P_2$ implique que P_1 divise V_2, ce qui est impossible puisque $\deg V_2 < d_1$. On en déduit que $V_1 = V_2 = 0$. En particulier, $\tilde{\omega}$ est racine de U_1, et ω est racine de $\tilde{U}_1$. Pour un ω fixé, U_1 étant de degré strictement inférieur à $(d_2 - 1)$, il existe au plus $(d_2 - 2)$ valeurs possibles pour $\tilde{\omega}$, ce qui prouve le résultat attendu. □

La preuve de ce dernier lemme est bien plus simple que les preuves des lemmes similaires précédemment utilisés. Elle repose exclusivement sur le comportement aléatoire de l'attribut choisi au moment du chiffrement, et plus du tout sur les attributs choisis et révélés lors de la génération des clés.

Le théorème donnant la sécurité globale de la variante du schéma dans le modèle générique des groupes avec couplage est le suivant :

Théorème 3.2 *Soit* $k < (\sqrt{p/(3l+6)} - n\,l - 2n - 4l - 8)/3$. *Dans le modèle générique des groupes avec couplage, l'avantage sur le problème IND-CPA de la variante du schéma de diffusion présenté en 3.5.3 d'un adversaire réalisant au plus* k *requêtes aux oracles est borné par :*

$$\frac{6\,k\,(l+2)\,(n\,l+2n+4l+3k+8)}{\left(1-\frac{1}{p-l}\right)\left(p-3\,(l+2)\,(n\,l+2n+4l+3k+8)^2\right)}.$$

3.6 Conclusion

Après une présentation des schémas classiques de diffusion, on a constaté l'absence d'un schéma efficace susceptible de traiter le cas d'une politique d'accès non pas basée sur une gestion individuelle des utilisateurs, mais sur une gestion via des groupes, définis par des attributs. En effet, soit le trafic généré pour le renouvellement des clés est extrêmement important (cas du schéma LKH), soit les messages chiffrés ne peuvent pas avoir une taille raisonnable (cas des schémas CS et SD), soit le mécanisme de déchiffrement nécessite des opérations trop coûteuses en temps (cas des schémas dérivés du chiffrement basé sur les attributs).

Le schéma proposé pour pallier à ce manque améliore significativement les performances en déchiffrement du seul schéma existant adapté à une gestion des utilisateurs via des groupes. Il permet de faire face à des coalitions de toutes tailles, les clés de déchiffrement sont relativement petites, les chiffrés sont plutôt courts. On peut même ajouter de nouveaux utilisateurs une fois le système déployé. Les seuls inconvénients de ce schéma portent sur la taille de la clé publique. On peut en fait diminuer significativement la taille de la clé publique, dans des usages raisonnables de ce schéma (en bornant le nombre d'attributs à imposer ou révoquer dans les messages chiffrés).

L'étude de ce schéma repose sur un modèle construit en fonction des politiques d'accès envisageables, différentes de celles des précédents schémas, mais sans changement majeur concernant l'objectif de sécurité. La preuve présentée repose sur le modèle générique présenté dans le chapitre précédent. Une preuve plus complexe aurait pu être construite par réduction sur une hypothèse plus conventionnelle, mais sans un réel gain en termes de sécurité.

Chapitre 4

Traçage de traîtres

Dans le chapitre concernant la diffusion (chapitre 3), l'objectif consiste à diffuser de manière confidentielle des données identiques vers de multiples destinataires. Ainsi, un utilisateur qui n'est pas destinataire ne peut pas obtenir ces données, et même une coalition d'utilisateurs non destinataires n'est pas en mesure de déchiffrer les données. Il existe cependant une vulnérabilité intrinsèque : les destinataires valides des messages peuvent déchiffrer les données et les transmettre à d'autres utilisateurs. Ils peuvent même divulguer leurs clés à ces utilisateurs, qui peuvent dès lors déchiffrer les données transmises comme s'ils en étaient destinataires.

On ne peut pas se protéger contre ces divulgations. Le traçage de traîtres consiste à lutter d'une manière indirecte contre ces pratiques, en identifiant la clé de déchiffrement utilisée. On peut dès lors identifier le responsable de la fuite des données vers des utilisateurs non-destinataires. Si une coalition de destinataires légitimes « mélange » les clés de déchiffrement dont elle dispose, il doit être possible d'identifier l'un des membres de la coalition. Des actions diverses peuvent alors être entreprises contre ce destinataire : il s'agit moins de « sanctionner » l'utilisateur identifié que de dissuader des utilisateurs a priori honnêtes de révéler leurs clés de déchiffrement.

L'objectif de cette étude est de proposer un schéma de traçage de traîtres sûr, même lorsque certaines mesures sont utilisées pour masquer les clés de déchiffrement utilisées. On considère donc un modèle de sécurité fort, où les données envoyées ne sont pas nécessairement déchiffrées, et où le traçage doit être indétectable. Le schéma proposé dans ce modèle utilise le marquage des données (*watermarking*) pour permettre l'identification d'un traître. En termes de performance, il a des chiffrés de taille constante, c'est-à-dire indépendante du nombre d'utilisateurs ou de la taille maximale de la coalition prévue. Par rapport aux autres schémas de traçage de traîtres à chiffrés de taille constante (non-sûrs dans le modèle présenté ici), il présente également l'avantage de permettre aux destinataires de déchiffrer les messages progressivement : dans les schémas précédents, on ne peut déchiffrer les données qu'en blocs de taille très importante.

4.1 Introduction au traçage de traîtres

4.1.1 Problématique concrète

La problématique du traçage de traîtres a été introduite dans [CFN94] et concerne des diffuseurs de contenus numériques protégés. Pour sécuriser la diffusion de ces contenus vers ses destinataires, un diffuseur va faire appel à un mécanisme de chiffrement. Ainsi, l'accès aux données claires ne peut être réalisé qu'avec un mécanisme de déchiffrement valide.

En revanche, un destinataire peut reconstituer les données claires à partir de son mécanisme de déchiffrement valide, et transmettre ces données à des tiers. Cette technique est théoriquement très convaincante, mais elle reste en pratique limitée. En effet, même avec des moyens de communication modernes, la diffusion à grande échelle nécessite une bande passante importante, et donc peu discrète. De plus, la rediffusion des contenus en clair implique de déchiffrer en permanence le flux en provenance de l'émetteur légitime. Un stratégie bien meilleure consiste pour un destinataire malhonnête à répliquer son mécanisme de déchiffrement. Il lui suffit en effet de confier des copies de son mécanisme de déchiffrement à des tiers, pour leur permettre d'accéder directement aux contenus. Le destinataire malhonnête n'a alors ni besoin de bande passante élevée, ni d'être en ligne en permanence.

Ce procédé est préjudiciable au diffuseur de contenus, sans qu'il ne lui soit possible de s'en protéger par le chiffrement : ses destinataires doivent nécessairement pouvoir accéder aux contenus, et il ne peut empêcher certains d'entre eux de divulguer leurs moyens de déchiffrement à des tiers (même si différents moyens technologiques peuvent être utilisés pour empêcher les destinataires d'accéder trop facilement à leurs mécanismes de déchiffrement).

Par contre, le diffuseur peut essayer d'identifier ses destinataires malhonnêtes. S'il y parvient, il pourra mettre en place des sanctions à leur encontre, le plus simple étant de les exclure de la diffusion des données. Ainsi, à défaut de les empêcher d'agir par des moyens cryptographiques, la menace que constitue ces sanctions dissuade les destinataires de divulguer leurs mécanismes de déchiffrement.

Dans le cadre d'une transmission à sens unique, du diffuseur vers l'ensemble des destinataires, le diffuseur n'a aucun retour d'information. L'identification de **traîtres**, c'est-à-dire de destinataires malhonnêtes, passe nécessairement par la capture d'un **décodeur pirate**, contenant un mécanisme de déchiffrement copié. La situation est en fait plus compliquée, puisque différents traîtres peuvent décider de former une coalition, de manière à « mélanger » leurs mécanismes de déchiffrement : le décodeur pirate résultant de leur association n'est alors formellement la copie d'aucun des décodeurs légitimes. À partir d'un tel décodeur pirate, il doit être possible de remonter à au moins l'un des traîtres qui a participé à sa conception : cette opération est appelée **traçage**.

Un mécanisme de traçage de traîtres peut être envisagé dans des applications de vente de contenus numériques, comme de la musique ou de la vidéo (télévision à péage), ou l'accès sécurisé à des ressources sur internet (encyclopédie, base de données...).

4.1.2 Quelques schémas existants

4.1.2.1 Distribution de clés de déchiffrement indépendantes

Le traçage de traîtres est une fonctionnalité évoluée : si la diffusion des données par le biais d'un chiffrement est une opération cryptographique plutôt classique, le traçage est peu courant. Comment modéliser ce mécanisme ?

Une première approche consiste à générer des clés de déchiffrement indépendantes, donc non-recombinables, et à donner à chacun des utilisateurs du système un sous-ensemble de ces clés de déchiffrement. Un mode de chiffrement adapté permet d'exiger des déchiffreurs la connaissance d'un ensemble de clés de déchiffrement, dont la répartition vérifie certaines propriétés. Cette approche, basée sur une répartition bien choisie de clés indépendantes, a été introduite et développée dans [CFN94, NP98, CFNP00].

Par rapport aux premiers schémas présentés dans [CFN94], la contribution essentielle de [NP98] est de remarquer qu'un décodeur pirate est inutilisable s'il ne déchiffre qu'une proportion minime des contenus diffusés. Alors que les premiers schémas permettaient un traçage de décodeurs pirates quelconques, [NP98] propose de nouveaux schémas, plus efficaces, traçant uniquement des décodeurs pirates déchiffrant des contenus avec une probabilité supérieure à un seuil clairement fixé.

Dans un premier temps, la procédure de traçage proposée dans tous ces schémas consistait à considérer que lorsqu'un décodeur pirate est capturé, on peut directement obtenir les clés de déchiffrement utilisées par ce décodeur pirate. En s'appuyant sur la distribution des clés de déchiffrement utilisées par le décodeur pirate, on peut alors identifier un traître probable.

Le principe de l'accès direct aux clés de déchiffrement utilisées par le décodeur pirate est cependant controversé. Le décodeur pirate peut être muni de contre-mesures diverses empêchant une lecture facile des clés, comme par exemple des techniques d'obfuscation de code dans le cas d'un déchiffrement logiciel. Pour contourner ces difficultés, une technique d'identification des clés de déchiffrement à partir de simples requêtes au décodeur pirate a été proposée dans [CFNP00], pour ces mêmes schémas : il s'agit d'envoyer des chiffrés partiellement erronés au décodeur pirate, et d'observer s'il est en mesure d'obtenir les messages associés. On est donc passé d'un modèle **en boîte ouverte** (*open-box*) à un modèle en **boîte noire** (*black-box*).

4.1.2.2 Clés disposant de propriétés particulières

Une approche totalement inverse s'est développée un peu plus tard, avec pour cible des protocoles asymétriques. Dans ces schémas, les clés de déchiffrement présentent une propriété particulière commune, et la recombinaison de clés est possible. Par contre, la recombinaison d'un nombre limité de clés de déchiffrement préserve non seulement cette propriété fondamentale, permettant le déchiffrement, mais aussi d'autres spécificités qui peuvent être testées, et qui permettent d'identifier un traître.

Le premier schéma de ce type a été proposé dans [KD98]. De nombreux autres sont apparus par la suite, avec notamment [BF99, MSK99, TSNZ03, MI04, BSW06, BW06]. Certains de ces schémas ont été mis en défaut, et c'est notamment le cas de [KD98, MSK99, TSNZ03].

On aborde pour l'exemple le schéma proposé dans [BF99]. Essentiellement, une clé de déchiffrement est associée à un vecteur appartenant à un hyperplan affine spécifique d'un espace vectoriel. L'ensemble des clés de déchiffrement est ainsi stable par combinaisons linéaires dont la somme des coefficients est égale à 1. Pour assurer le traçage de coalitions allant jusqu'à t traîtres, la dimension de l'espace vectoriel et les clés de déchiffrement transmises aux utilisateurs du système sont choisis de manière appropriée : les espaces vectoriels issus des recombinaisons de t clés de déchiffrement ont des intersections de dimension nulle si les ensembles de clés sont disjoints. De plus, une technique de codes correcteurs d'erreurs permet de reconstituer une clé de déchiffrement de l'un des traîtres à partir de la clé de déchiffrement du décodeur pirate.

Le traçage en boîte noire est plus délicat. En fait, le protocole dispose d'une méthode de confirmation de culpabilité, en boîte noire, c'est-à-dire qu'à partir d'un ensemble suspect composé d'au plus t membres, il est possible d'envoyer un message chiffré tel que seule une combinaison linéaire des clés de déchiffrement de ces membres permet de retrouver le message clair associé. On peut donc vérifier que le décodeur pirate a été conçu par une coalition contenue dans l'ensemble ciblé. Cette technique peut être utilisée en méthode de traçage, mais avec une forte inefficacité, puisqu'il faut tester tous les sous-ensembles de t utilisateurs parmi n.

4.1.2.3 Répartition de clés recombinables

Un principe particulièrement attractif a été développé dans [KY02]. L'idée consiste à considérer un schéma de traçage de traîtres limité à deux utilisateurs : le faible nombre d'utilisateurs permet des constructions simples et efficaces, même dans un cadre asymétrique. Pour généraliser à un grand nombre d'utilisateurs, on génère de nombreuses instances de ce schéma spécifique, en choisissant subtilement la répartition des clés de déchiffrement liées à chaque instance.

Concrètement, pour chaque instance, un utilisateur reçoit l'une des deux clés de déchiffrement associée à cette instance. Le chiffrement utilise successivement toutes les instances. Le traçage consiste à identifier pour chaque instance la clé de déchiffrement utilisée, ce qui peut se faire très facilement, et même dans un modèle en boîte noire : l'identification d'un traître à partir de l'ensemble des clés de déchiffrement détectées découle de la répartition des clés.

L'intérêt fondamental de ces schémas est qu'ils présentent un taux de chiffrement constant, c'est-à-dire que le ratio de tailles entre un message clair et son chiffré associé ne dépend ni du nombre d'utilisateurs, ni du nombre maximal de traîtres considéré.

Plus récemment, cette construction a été optimisée dans [CPP05], en utilisant des couplages. Outre un ratio entre les tailles des messages chiffrés et des messages clairs

presqu'égal à 1, ce dernier schéma présente l'intérêt d'être partiellement traçable publiquement : l'essentiel de la procédure de traçage peut être déléguée à une entité n'utilisant que la clé publique du schéma.

4.1.3 Des modèles divers

Même avec une problématique fixée, le modèle à adopter pour décrire un mécanisme de traçage de traîtres n'est pas clair. Il existe de nombreuses manière d'appréhender le comportement d'un décodeur pirate, et différentes visions du traçage.

4.1.3.1 Traçage en boîte ouverte ou en boîte noire

Dans un premier temps, on a mis en évidence précédemment la différence entre le traçage en boîte ouverte et le traçage en boîte noire. En fait, les schémas récents utilisent quasi-exclusivement le traçage en boîte noire : le traceur envoie des requêtes au décodeur pirate, et identifie un traître à partir de ses réponses aux requêtes.

4.1.3.2 Taux de réponse du décodeur pirate

Il existe d'autres différences de modèles assez significatives. Puisque les schémas récents s'appuient sur des mécanismes de traçage en boîte noire, qui nécessitent des réponses de la part du décodeur pirate, il est important de savoir dans quelle mesure le décodeur pirate risque de ne pas répondre aux requêtes qui lui sont envoyées.

Dans certains schémas (comme ceux proposés dans [CFN94]), il suffit que le décodeur pirate réponde occasionnellement pour que le traçage soit possible.

D'autres (ceux de [NP98] par exemple) font appel à un taux minimum de réponse du décodeur pirate : si le décodeur pirate répond aux requêtes avec une probabilité supérieure à ce taux, le traçage est possible, dans le cas contraire, le schéma ne permet pas le traçage. Considérer ce type de comportement est raisonnable, dans la mesure où un décodeur pirate qui déchiffre trop rarement n'est pas exploitable, et ne présente donc pas une menace. Comme cela permet des schémas significativement meilleurs que dans le cas précédent, cette optimisation semble pertinente.

Enfin, certains schémas supposent que le décodeur pirate répond en permanence à toutes les requêtes qui lui sont faites (c'est le cas en première approximation des schémas basés sur une répartition de clés recombinables, comme [KY02, CPP05]). Comme un décodeur pirate peut aisément sortir de ce type de comportement, un ajustement de ces schémas est nécessaire pour tenir compte d'un cadre plus réaliste. Ainsi, dans le cas des schémas basés sur une répartition de clés recombinables, un mécanisme de mélange est utilisé pour forcer le décodeur pirate à déchiffrer de manière homogène des messages apportant au traceur des informations complémentaires.

4.1.3.3 Mémoire du décodeur pirate

Dans [KY01a, KY01b], un décodeur pirate peut disposer d'une mémoire, et l'utiliser pour chercher à tromper la procédure de traçage. Une telle hypothèse semble plutôt raisonnable, mais dans de nombreux schémas, la procédure de traçage nécessite que le décodeur pirate ne mémorise pas les requêtes qui lui sont transmises.

Bien souvent, la justification de cette hypothèse forte consiste à supposer que le décodeur pirate est capturé sous une forme logicielle, et qu'on peut par conséquent en réaliser de nombreuses copies. Il suffit alors de n'envoyer qu'une unique requête à chacune des copies du logiciel tracé pour simuler un décodeur pirate ne disposant pas de mémoire.

4.1.3.4 Caractère abrupt du décodeur pirate

Toujours dans [KY01a, KY01b], un décodeur pirate peut être abrupt, c'est-à-dire qu'il peut chercher à s'arrêter brutalement lorsqu'une procédure de traçage est conduite contre lui. Concrètement, le décodeur pirate essaie de savoir si les requêtes qu'il reçoit sont issues d'un envoi normal de données, ou si au contraire il s'agit d'une procédure de traçage. S'il est convaincu qu'il s'agit du second cas, le décodeur pirate peut utiliser des mécanismes de protection consistant à effacer immédiatement ses clés.

Naturellement, un décodeur pirate abrupt peut tirer fortement parti d'une mémoire, de manière à identifier une procédure de traçage non pas à partir d'une seule requête, mais à partir d'une succession de requêtes.

Le caractère abrupt d'un décodeur pirate présente cependant un risque : un émetteur peut diffuser largement des messages mal formés, simulant ainsi une procédure de traçage fictive. Ainsi, les décodeurs pirates abrupts peuvent réagir, et effacer leurs clés, alors même qu'ils ne sont pas en danger !

4.1.4 Performances et fonctionnalités des schémas

La performance des schémas est évaluée en grande partie sur le taux de chiffrement, qui correspond à la taille d'un chiffré divisé par la taille du message clair correspondant. Cet élément est fondamental puisqu'il mesure la bande passante nécessaire pour transmettre des données. Comme on l'a déjà mentionné, les schémas utilisant une répartition de clés recombinables, comme [KY02, CPP05], présentent un taux de chiffrement constant.

La contrepartie de ces schémas est que les messages chiffrés sont très longs, et qu'il est nécessaire de les déchiffrer complètement avant d'obtenir le moindre bit d'information sur le message clair associé. Cela peut être un problème, puisque ce phénomène peut induire des délais importants dans la transmission d'un contenu : il est impossible de déchiffrer progressivement ce contenu. En dehors du taux de chiffrement, la taille des chiffrés, mais aussi le coût calculatoire du déchiffrement et la taille d'une clé de déchiffrement sont des éléments à prendre en compte pour évaluer l'efficacité d'un schéma de traçage de traîtres.

En dehors des performances quantifiables de cette nature, certaines fonctionnalités additionnelles peuvent être précieuses. On peut particulièrement citer le cas du traçage public, évoqué dans [CPP05], mais pas complètement réalisé : cette fonctionnalité consiste à autoriser la délégation de traçage, c'est-à-dire qu'un tiers réalise le traçage, sans disposer d'aucun privilège et sans pouvoir lui-même échapper à un traçage. Dans [CPP05], une partie du traçage peut être déléguée, tandis que le reste doit impérativement être réalisé par l'autorité (l'accès à un oracle de traçage permet en fait de construire un décodeur pirate échappant au traçage).

On peut également mentionner la possibilité donnée dans certains schémas de tracer des coalitions de taille quelconque, comme c'est le cas dans [BSW06, BW06]. Malheureusement, ces deux schémas ont une efficacité limitée en comparaison de schémas destinés à tracer des coalitions de taille restreinte.

4.1.5 Contribution

Il existe de nombreuses directions pour tenter d'améliorer les schémas de traçage de traîtres existants : utiliser un modèle de sécurité plus fort, améliorer l'efficacité du schéma, fournir des fonctionnalités additionnelles... Le but principal de ce chapitre est de se concentrer sur les deux premiers points, et de concilier un modèle de sécurité fort avec de bonnes performances.

Pour cela, on présente en partie 4.2 un modèle pour les décodeurs pirates qui se rapproche du modèle présenté dans [KY01a], tout en étant légèrement plus fort. Comme dans [KY01a], on considère qu'un décodeur pirate est abrupt et dispose d'une mémoire. En revanche, il peut répondre aux requêtes avec un taux de réponse minimum fixé, alors que le modèle précédent suppose des réponses systématiques. Sur ce dernier aspect, le modèle utilisé se rapproche de celui de [NP98].

La construction d'un schéma de traçage de traîtres sûr dans ce modèle et avec une bonne efficacité s'appuie sur la stratégie élaborée par [KY02] et présentée en sous-section 4.1.2.3. On construit d'abord un schéma restreint à deux utilisateurs, et on le généralise par l'emploi d'un code adapté.

La construction du schéma élémentaire (partie 4.3) s'appuie notamment sur [KY01a] et utilise un schéma de tatouage (*watermarking*), comme par exemple le schéma donné dans [FT99]. La généralisation à un nombre d'utilisateurs bien plus grand nécessite un code sûr contre des coalitions avec effacements (partie 4.4). On donne une définition adaptée à notre cadre d'étude, et à défaut de pouvoir utiliser des codes pré-existants, on construit une famille spécifique de codes sûre contre des coalitions avec effacement.

Le schéma global s'appuie sur ces deux composants, et sa sécurité s'en déduit directement (partie 4.5). Le schéma présente un taux de chiffrement constant, ce qui est nouveau dans un tel modèle. Les messages chiffrés sont toujours très longs, mais à la différence de [KY02, CPP05], ils peuvent être déchiffrés au fur et à mesure de la réception des morceaux de chiffrés. Le délai induit est ainsi considérablement réduit.

4.1.6 Préliminaires

Dans tout ce chapitre, en particulier dans les définitions, les théorèmes et les preuves, on utilise implicitement un paramètre λ : tous les algorithmes sont considérés polynomiaux en temps en λ. Le niveau de sécurité dépend naturellement de ce paramètre λ, dans la mesure où il représente les limites d'un adversaire dans le modèle de sécurité.

4.2 Un modèle pour les schémas et les décodeurs pirates

4.2.1 Schémas de traçage de traîtres

La définition suivante pour un schéma de traçage de traîtres est similaire à des définitions plus anciennes de schémas basés sur une traçabilité en boîte noire, comme celui de [BSW06].

Définition 4.1 (Schéma de traçage de traîtres) *Un schéma de traçage de traîtres consiste en quatre algorithmes :*

- *l'**initialisation** est un algorithme probabiliste qui reçoit le nombre n d'utilisateurs du système et qui construit une clé de chiffrement ek, une clé de traçage tk, et n clés de déchiffrement : $dk_1, \ldots, dk_n$;*
- *le **chiffrement** est un algorithme probabiliste E qui reçoit un message m et qui construit un chiffré c, en utilisant la clé de chiffrement ek ;*
- *le **déchiffrement** est un algorithme déterministe D qui reçoit un chiffré c et retrouve le message m associé, en utilisant une clé de déchiffrement dk_u. Pour tout message m, et tout utilisateur u, il vérifie $D_{dk_u}(E_{ek}(m)) = m$;*
- *le **traçage** est un algorithme probabiliste T qui construit des requêtes adressées à un décodeur pirate, vu comme un oracle de déchiffrement, en utilisant la clé de traçage tk. Il retourne un ensemble d'utilisateurs, calculé à partir des réponses données par le décodeur pirate à ses requêtes.*

Pour utiliser ce schéma de traçage de traîtres, un fournisseur de contenu commence par lancer l'initialisation qui construit les clés pour l'émetteur, pour le traceur et pour tous les utilisateurs du schéma. Le nombre maximum d'utilisateurs, n, doit être défini à l'avance, même si ces utilisateurs ne sont pas connus : lorsqu'un nouvel utilisateur rejoint le système, il reçoit une clé de déchiffrement calculée initialement.

Les procédures de chiffrement et de déchiffrement permettent à l'émetteur de diffuser des données aux utilisateurs.

La procédure de traçage interagit avec un décodeur pirate : elle construit des requêtes qui sont traitées par le décodeur. Ces requêtes peuvent être des messages chiffrés valides, ou des messages chiffrés altérés. Dans ce dernier cas, seuls certains utilisateurs sont en mesure d'obtenir le message clair associé au message chiffré. Les autres utilisateurs obtiennent des résultats complètement différents.

Cette définition peut être utilisée pour de nombreux schémas de traçage de traîtres. Lorsqu'on s'intéresse aux schémas à clé publique, la clé de chiffrement *ek* est publique.

De même, lorsqu'on modélise un schéma dans lequel le traçage peut être délégué, la clé de traçage tk est considérée comme publique.

4.2.2 Décodeurs pirates

On s'intéresse désormais aux décodeurs pirates. Un décodeur pirate est construit par une coalition de traîtres, et contient un ensemble de clés de déchiffrement. En mode nominal, le décodeur pirate reçoit des messages chiffrés valides et déchiffre ces messages. Par contre, lorsqu'il a été capturé, il est interrogé par une procédure de traçage et reçoit des requêtes visant à extraire de l'information : dans ce second cas, il doit dans la mesure du possible chercher à protéger les clés de déchiffrement qu'il utilise.

On considère un décodeur pirate dans le contexte le plus large possible. Dans un premier temps, on suppose qu'un décodeur pirate est doté d'une mémoire : il peut donc enregistrer toutes les requêtes qui lui ont été soumises, et les réponses qu'il a faites à ses requêtes. Ceci peut lui être particulièrement utile, dans la mesure où un décodeur pirate ne sait pas dans quel contexte (réception de messages valides ou traçage) il se trouve : seule une étude des requêtes reçues peut lui permettre d'identifier une procédure de traçage et de modifier son comportement. Avec l'accès à une mémoire, le décodeur pirate peut analyser la succession des requêtes qu'il reçoit. Dans certains cas, comme par exemple dans [FT99], les requêtes successives d'une procédure de traçage s'adaptent aux réponses du décodeur pirate : ce dernier peut donc vérifier si les requêtes qu'il reçoit s'adaptent à ses réponses ou pas.

Dans le cas où le décodeur pirate détecte une procédure de traçage en cours contre lui, on lui donne la possibilité de prendre des mesures de protection, comme l'effacement instantané de ses clés de déchiffrement. Cette mesure doit cependant être prise uniquement lorsque la procédure de traçage est détectée de manière fiable par le décodeur pirate : dans certains schémas (comme par exemple [MI04] face à des décodeurs pirates sans mémoire), l'émetteur tente de déclencher de telles réactions pendant l'émission, pour faire en sorte d'éliminer les décodeurs pirates.

Enfin, on permet au décodeur pirate de refuser de répondre à certaines requêtes, même quand il s'agit de messages chiffrés complètement authentiques, de la même manière que dans [NP98]. Dans ce modèle, un décodeur pirate peut refuser de déchiffrer des messages chiffrés valides avec une probabilité donnée, ou simplement refuser de déchiffrer des messages chiffrés ayant une propriété particulière. Dans des applications multimédia, la perte d'une partie du contenu peut ne pas être préjudiciable, et si cela peut éviter le traçage, c'est un compromis raisonnable pour un décodeur pirate.

Globalement, avec ces contraintes, on demande à un décodeur pirate de répondre avec un taux minimum de réponse lorsqu'il reçoit exclusivement des messages chiffrés authentiques. Dès lors qu'il identifie une requête de traçage, il peut refuser de déchiffrer toute requête ultérieure, ce qui correspond au comportement d'un décodeur abrupt.[1]

[1] Face à des décodeurs pirates abruptes, mais sans mémoire, on obtient de l'information lorsque le décodeur pirate s'arrête : la requête ayant déclenché l'arrêt du décodeur est nécessairement distinguable

Définition 4.2 (Décodeur pirate fort) *Un décodeur pirate fort $\mathcal{P}$ est un algorithme probabiliste initialisé avec les clés de déchiffrement distribuées à une coalition C d'utilisateurs $\{dk_u \,/\, u \in C\}$. Il reçoit des requêtes : pour chaque requête q_i il calcule une réponse a_i, éventuellement vide (notée $\perp$ dans ce cas). Chaque réponse a_i peut dépendre des requêtes passées $(q_j)_{j \leqslant i}$ et des réponses précédemment renvoyées $(a_j)_{j<i}$.*

Dans cette définition, on suppose que le décodeur pirate fort dispose de l'ensemble des clés de déchiffrement des membres de la coalition. En fait, il est probable que la coalition calcule une « nouvelle » clé de déchiffrement, voire un nouveau mécanisme de déchiffrement pour éviter de confier toutes les clés de déchiffrement de ses membres à des décodeurs pirates. Mais dans le modèle, on considère que ce calcul peut être fait par le décodeur pirate à partir des clés de déchiffrement qui lui ont été initialement confiées. Le modèle est donc raisonnable.

4.2.3 Sécurité d'un schéma de traçage de traîtres

On peut désormais définir le jeu consistant à tracer un décodeur pirate. Ce jeu sera le cœur de la définition de sécurité des schémas de traçage de traîtres.

Définition 4.3 (Jeu de traçage) *Le jeu de traçage fait intervenir un environnement Ω et un décodeur pirate fort $\mathcal{P}$ d'un schéma de traçage de traîtres :*

- *Ω choisit n, et génère les clés ek, tk, $dk_1, \ldots, dk_n$;*
- *$\mathcal{P}$ choisit une coalition $C \subset \{1, \ldots, n\}$, et révèle cette coalition à Ω ;*
- *Ω transmets les clés $\{dk_u \,/\, u \in C\}$ à $\mathcal{P}$;*
- *Ω réalise la procédure de traçage T du schéma de traçage de traîtres sur $\mathcal{P}$.*

Le décodeur pirate $\mathcal{P}$ gagne le jeu si l'ensemble d'utilisateurs donné par la procédure de traçage T est vide ou non-inclus dans C.

Évidemment, un décodeur pirate ne peut être tracé s'il ne répond à aucune requête. La procédure de traçage n'est donc censée donner des résultats corrects que lorsque le décodeur pirate répond aux requêtes valides avec un taux supérieur à un seuil donné. En-dessous de ce seuil, le décodeur pirate est jugé inexploitable, et le traçage est donc inutile. On définit alors la sécurité d'un schéma de traçage de traîtres, en tenant compte de ces restrictions, par la capacité de la procédure de traçage à identifier un traître à partir d'un décodeur pirate fort :

Définition 4.4 (Sécurité d'un schéma) *Un schéma de traçage de traîtres est dit (p, t, α)-sûr si un décodeur pirate fort qui déchiffre des messages chiffrés valides avec une probabilité supérieure ou égale à α gagne le jeu de traçage sur ce schéma avec une probabilité au plus p, lorsque la coalition qu'il choisit est de taille au plus égale à t.*

---

d'un message chiffré authentique, pour ce décodeur pirate. Ce n'est plus le cas lorsque les décodeurs pirates disposent de mémoire ou peuvent nominalement refuser de répondre à certaines requêtes formées de messages chiffrés authentiques : du point de vue du traceur, il n'est pas possible de déterminer avec certitude quel message de traçage a provoqué l'arrêt d'un décodeur pirate.

Remarque 4.1 *Dans les définitions précédentes, on s'intéresse uniquement à la capacité de tracer. La sécurité d'un schéma de traçage de traîtres en tant que schéma de chiffrement (* « est-ce qu'il est possible de distinguer un chiffré d'un autre, sans connaître aucune clé de déchiffrement ? »*) n'est pas directement abordée. Dans les schémas de traçage de traîtres cités, la sécurité en tant que mécanismes de chiffrement est bien assurée.*

4.2.4 Sécurité de certains schémas dans ce modèle

Les schémas de traçage de traîtres sont traditionnellement placés dans des modèles plus faibles, et ne supportent pas le modèle de sécurité décrit en section 4.2.3.

Ainsi, les schémas basés sur des propriétés algébriques, comme [BF99, KD98], ont été considérés dans un modèle intermédiaire dans [KY01b] : ces schémas ne peuvent pas être sûrs dans un tel modèle lorsque le nombre des traîtres évolue plus que logarithmiquement en fonction du nombre d'utilisateurs. Même en utilisant des couplages, ces contraintes ne semblent pas pouvoir être contournées : dans [BSW06], les auteurs expliquent eux-mêmes que leur schéma n'est sûr que dans un modèle faible.

À titre d'illustration, la procédure de confirmation en boîte noire proposée dans [BF99] consiste en l'envoi d'un pseudo message chiffré, qui n'est réellement indistinguable d'un message chiffré authentique que pour l'ensemble ciblé par la confirmation. Si cet ensemble ne contient pas entièrement la coalition (et c'est typiquement le cas lorsqu'on transforme cette procédure de confirmation en procédure de traçage), le décodeur pirate peut obtenir des messages clairs différents en utilisant les clés de déchiffrement des membres de la coalition. Il ne s'agit donc pas d'un chiffré valide, et le décodeur pirate peut se bloquer.

Dans les schémas basés sur une répartition bien choisie de clés de déchiffrement indépendantes, comme [CFN94, NP98], l'extraction des clés utilisées par le décodeur pirate dans un modèle en boîte noire nécessite de considérer seulement des décodeurs pirates sans mémoire. Cependant, ces schémas peuvent être modifiés de manière analogue à [KY01a], et ils deviennent alors (p, t, α)-sûrs, lorsque le seuil α est suffisamment élevé.

Le cas des schémas présentés dans [FT99] occupe une place assez singulière. Avec l'utilisation du marquage (*watermarking*), ces schémas ont l'avantage de présenter des messages de traçages parfaitement indistinguables de messages chiffrés valides, ce qui est généralement l'obstacle majeur pour assurer une sécurité dans un modèle fort. En revanche, pour être efficaces, les procédures de traçage sont adaptatives, c'est-à-dire que la succession des requêtes envoyées dépend des réponses du décodeur pirate. Un décodeur pirate avec mémoire peut identifier ce genre de comportement, et par conséquent se bloquer. De plus, les messages permettant d'identifier précisément un traître, lorsque deux utilisateurs sont suspects, ont une structure particulière : si un décodeur refuse systématiquement de répondre à ce genre de requêtes, il refuse une proportion relativement faible de messages, mais empêche toute procédure de traçage d'arriver à son terme.

Le cas des schémas à répartition de clés recombinables, comme [CPP05, KY02], sera étudié en détails ultérieurement (paragraphe 4.4.3.1). Dans une première formulation, ils ont besoin de décodeurs pirates qui répondent systématiquement à toutes les requêtes, et les messages de traçage sont distinguables de messages chiffrés valides. L'utilisation du marquage de données permet de rendre indistinguables les messages de traçage et les chiffrés valides. En revanche, cette modification est incompatible avec l'utilisation de codes sûrs contre des coalitions, qui prend en compte la possibilité pour un décodeur pirate de refuser de déchiffrer certaines requêtes. Ces schémas ne sont donc pas sûrs dans un modèle fort.

Le seul schéma qui présente une sécurité prouvée dans un modèle proche est proposé dans [KY01a]. Ce schéma ne prend pas en compte la possibilité pour un décodeur pirate de refuser de répondre à certaines requêtes, mais pour des taux de réponse minimum raisonnables, le schéma reste sûr. Il a par contre l'inconvénient d'être peu efficace : le taux de chiffrement est important, et c'est également le cas des schémas [CFN94, NP98] qui peuvent être adaptés à des décodeurs pirates forts.

Un schéma de traçage de traîtres sûr face à des décodeurs pirates forts, et présentant cependant un taux de chiffrement indépendant du nombre d'utilisateurs et de la taille maximale des coalitions considérée semble donc une amélioration significative.

4.3 Schéma élémentaire pour deux utilisateurs

Pour répondre à cet objectif de construction d'un schéma de traçage de traîtres présentant à la fois une sécurité face à des décodeurs pirates forts et un taux de chiffrement constant, on utilise une stratégie similaire à celle présentée dans [KY02].

Dans un premier temps, on définit un schéma élémentaire, pour deux utilisateurs, avec une sécurité face à des décodeurs pirates forts. C'est le cadre de cette partie. Par la suite (partie 4.4), on cherchera un moyen de recouper des informations issues de diverses instances de ce schéma restreint à deux utilisateurs.

4.3.1 Utilisation du marquage

4.3.1.1 Nécessité du marquage

- Lors d'une procédure de traçage, pour apporter de l'information, une requête doit nécessairement pouvoir mener à différentes réponses possibles, en fonction de la clé de déchiffrement utilisée.
- À l'inverse, un message chiffré valide conduit systématiquement à une même réponse à partir de n'importe quelle clé de déchiffrement.

Ces deux propriétés antagonistes laissent penser qu'il est impossible de construire des messages de traçage et des messages chiffrés valides qui soient indistinguables. En fait, on peut relâcher la seconde propriété : on cherche à transmettre des données similaires à tous les utilisateurs, mais pas forcément des données identiques.

Dans la lignée de [FT99, KY01a], la solution consiste à utiliser du marquage (*watermarking*) des données : à partir d'un contenu à transmettre, un émetteur va générer deux variantes du contenu initial, mais avec quelques légères modifications. Pour de nombreux types de données, ces petites différences sont imperceptibles, et les utilisateurs reçoivent donc bien le contenu prévu. Par contre, l'émetteur est en mesure de faire la différence entre les variantes qu'il a transmises.

L'utilisation du marquage implique qu'on ne travaille que sur des contenus pouvant être légèrement modifiés sans que cela ne pose de problèmes d'utilisation. C'est le cas de la plupart des contenus multimédia (musique, images, vidéos...). Par contre, on ne peut pas utiliser du marquage pour envoyer des clés cryptographiques : on ne peut donc pas faire du chiffrement hybride directement.

4.3.1.2 Définition

Pour donner une définition cryptographique au marquage, on s'appuie sur le formalisme et les idées développées dans [CKLS96, EKK99] :

Définition 4.5 (Schéma de marquage) *Un schéma de marquage (pour deux utilisateurs) consiste en trois algorithmes :*

- *la* ***fonction d'acceptation*** *est un algorithme déterministe* A *qui prend en entrée un couple de messages* (m, m') *et répond par un booléen* $\mathsf{A}(m, m') \in \{vrai, faux\}$ *;*
- *la* ***fonction de génération*** *est un algorithme probabiliste* G *qui prend en entrée un message* m *et construit deux variantes* m_1 *et* m_2 *de ce message, vérifiant* $\mathsf{A}(m, m_1) = \mathsf{A}(m, m_2) = vrai$ *;*
- *la* ***fonction de détection*** *est un algorithme déterministe* D *qui prend en entrée un quadruplet de messages* (m, m_1, m_2, m')*, où* m_1 *et* m_2 *sont des variantes du message* m*, et répond avec un indice* $u' \in \{0, 1, 2\}$*.*

La fonction d'acceptation A est une manière de déterminer si deux messages ont un contenu proche ou non. Si $\mathsf{A}(m, m')$ est vrai, alors on considère que les messages m et m' sont proches l'un de l'autre, et que par conséquent m' pourrait être utilisé à la place de m. Si une distance existe sur l'espace des messages, alors une bonne définition de la fonction d'acceptation A pourrait être : $\mathsf{A}(m, m')$ = vrai si et seulement si la distance entre m et m' est inférieure à un paramètre δ.

La fonction de génération renvoie deux variantes d'un message donné, chacune de ces variantes étant acceptable pour le message d'origine. À partir d'un message m, de deux variantes de m générées ensemble m_1 et m_2, et d'un dernier message m', la fonction de détection identifie la variante (m_1 ou m_2) à partir de laquelle m' pourrait avoir été construit. Elle répond 1 pour m_1, 2 pour m_2, ou 0 en cas d'échec.

4.3.1.3 Sécurité

Un adversaire d'un schéma de marquage reçoit une variante m_u issue d'un message m, et construit un message m'. Son but est que le message m' construit soit acceptable

pour m, et puisse donc légitimement servir à sa place, tout en empêchant la fonction de détection d'identifier quelle variante de m a été utilisée pour construire m'.

Définition 4.6 (Adversaire d'un schéma de marquage) *Un adversaire $\mathcal{A}$ d'un schéma de marquage est un algorithme probabiliste, disposant des fonctions* A, G *et* D *d'un schéma de marquage. Il reçoit une variante m_u d'un message m inconnu, et calcule un message m'.*

On peut désormais donner le jeu de détection, et définir la sécurité d'un schéma de marquage :

Définition 4.7 (Sécurité d'un schéma de marquage) *Le jeu de détection fait intervenir un environnement Ω et un adversaire $\mathcal{A}$ d'un schéma de marquage* $(\mathsf{A}, \mathsf{G}, \mathsf{D})$ *sur un message m :*

- *Ω calcule un couple de variantes $(m_1, m_2) = \mathsf{G}(m)$;*
- *Ω choisit aléatoirement $u \in \{1, 2\}$ et transmet m_u à $\mathcal{A}$;*
- *$\mathcal{A}$ répond avec un message m' ;*
- *Ω calcule $u' = \mathsf{D}(m', m, m_1, m_2)$.*

L'adversaire $\mathcal{A}$ gagne le jeu s'il y a une mauvaise détection ($u' = \bar{u}$) ou si la détection échoue ($u' = 0$ avec $\mathsf{A}(m, m') = true$).

Un schéma de marquage $(\mathsf{A}, \mathsf{G}, \mathsf{D})$ *est dit (p_w, q_w)-sûr si pour tout message m et pour tout adversaire $\mathcal{A}$:*

- *la probabilité de mauvaise détection dans le jeu est majorée par p_w ;*
- *la probabilité d'échec de la détection dans le jeu est majorée par q_w.*

En fait, il y a mauvaise détection lorsque l'algorithme de détection incrimine la variante qui n'est pas connue de l'adversaire. Quel que soit le contexte, l'adversaire gagne le jeu dans ces conditions. La détection échoue lorsque l'algorithme de détection n'arrive pas à trancher entre les deux variantes, et que le message m' produit par l'adversaire est acceptable pour m.

Le cas où l'algorithme de détection n'arrive pas à trancher entre les deux variantes, alors que le message m' n'est pas acceptable pour m, n'est pas considéré comme une victoire de l'adversaire. Sinon, il suffit à l'adversaire de répondre aléatoirement et indépendamment de la variante m_u reçue pour obtenir une probabilité de gain non négligeable. Cette stratégie n'est pas révélatrice d'une faiblesse du schéma de traçage, puisque le contenu du message à transmettre est perdu.

4.3.2 Définition du schéma

4.3.2.1 Schéma de chiffrement

En dehors d'un schéma de marquage sûr, la définition du schéma élémentaire de traçage de traîtres nécessite un schéma de chiffrement sûr dans le modèle LoR-CPA :

Définition 4.8 (Sécurité LoR-CPA) *Le jeu LoR (Left or Right) fait intervenir un environnement* Ω *et un adversaire* $\mathcal{A}$ *d'un schéma de chiffrement constitué d'une fonction d'initialisation* I*, d'une fonction de chiffrement* E*, et d'une fonction de déchiffrement* D *:*

- Ω *choisit aléatoirement* $v \in \{1,2\}$ *et génère un couple de clés* (ek, dk) *avec* I *;*
- Ω *transmet* ek *à* $\mathcal{A}$ *(uniquement dans le cadre d'un schéma à clé publique) ;*
- $\mathcal{A}$ *envoie des paires de messages* (m_1, m_2) *à* Ω *qui répond avec* $\mathsf{E}_{ek}(m_v)$ *;*
- $\mathcal{A}$ *répond avec un élément* $v' \in \{1,2\}$*.*

L'adversaire $\mathcal{A}$ *gagne le jeu si* $v' = v$*.*

Le schéma de chiffrement est dit adv_e*-LoR-CPA sûr si pour tout adversaire* $\mathcal{A}$*,*

$$\big|\ \Pr[\mathcal{A} \text{ gagne le jeu LoR}] - \Pr[\mathcal{A} \text{ perd le jeu LoR}]\ \big| \leqslant adv_e.$$

Ce modèle est plus exigeant qu'un modèle IND-CPA classique : au lieu d'une requête unique où l'adversaire envoie un couple de messages et où l'environnement chiffre l'un de ces deux messages, l'adversaire dispose d'autant de requêtes qu'il le souhaite, l'environnement chiffrant toujours le message dans la même position. Ce modèle a été notamment utilisé dans [BDJR97] dans le contexte d'un chiffrement symétrique.

4.3.2.2 Schéma de traçage de traîtres

On peut désormais définir un schéma de traçage de traîtres pour deux utilisateurs s'appuyant sur un schéma de marquage et un schéma de chiffrement :

Définition 4.9 (Schéma élémentaire de traçage de traîtres) *Soit* $(\mathsf{A}, \mathsf{G}, \mathsf{D})$ *un schéma de marquage* (p_w, q_w)*-sûr, soit* $(\mathsf{I}, \mathsf{E}, \mathsf{D})$ *un schéma de chiffrement* adv_e*-LoR-CPA sûr. Le schéma élémentaire de traçage de traîtres s'appuyant sur ces schémas est défini par les algorithmes suivants :*

- ***Initialisation*** *: deux paires de clés,* (ek_1, dk_1) *et* (ek_2, dk_2)*, sont générées en utilisant l'algorithme* I*. La clé de chiffrement est* $ek = (ek_1, ek_2)$*. Les clés de déchiffrement sont* dk_1 *et* dk_2*. La clé de traçage est* $tk = ek$*.*
- ***Chiffrement*** *: le chiffré d'un message* m *est* $E_{ek}(m) = (\mathsf{E}_{ek_1}(m_{\sigma(1)}), \mathsf{E}_{ek_2}(m_{\sigma(2)}))$*, où* $(m_1, m_2) = \mathsf{G}(m)$ *et* σ *est une permutation sur* $\{1,2\}$ *choisie aléatoirement.*
- ***Déchiffrement*** *: l'utilisateur* u *obtient un message clair acceptable* $m_{\sigma(u)}$ *en utilisant sa clé de déchiffrement* dk_u *sur la partie adéquate du chiffré.*
- ***Traçage*** *: l'algorithme de traçage consiste à envoyer des messages chiffrés authentiques jusqu'à ce que le décodeur pirate ait répondu* N_p *fois avec un message acceptable pour le contenu envoyé. Pour chacune des réponses du décodeur pirate, l'algorithme calcule* $u' = \sigma(\mathsf{D}(m, m_1, m_2, m'))$*. L'algorithme répond alors avec l'élément le plus fréquemment trouvé dans* $\{1,2\}$ *(en cas d'égalité, le choix est fait aléatoirement). La valeur de* N_p *dépend du paramètre* $p \in]2adv_e, 1[$ *:*

$$N_p = \max\Big(3\left\lceil \frac{\log(p/2 - adv_e)}{\log(8\max(p_w, q_w))} \right\rceil, 1\Big).$$

Remarque 4.2 *La procédure de traçage s'applique pour n'importe quel décodeur pirate qui répond aux requêtes valides avec une probabilité supérieure à n'importe quel seuil* $\alpha > 0$. *Plus le seuil est bas, plus la procédure sera longue, puisqu'il faudra de l'ordre de* (N_p/α) *requêtes pour obtenir* N_p *réponses acceptables.*

4.3.3 Preuve de sécurité

Théorème 4.1 *Soit* $(\mathsf{I}, \mathsf{E}, \mathsf{D})$ *un schéma de chiffrement* adv_e*-LoR-CPA sûr, soit* $(\mathsf{A}, \mathsf{G}, \mathsf{D})$ *un schéma de marquage* (p_w, q_w)*-sûr. Pour tout* $p > 2adv_e$*, pour tout* $\alpha > 0$*, le schéma de traçage de traîtres à deux utilisateurs présenté en définition 4.9 est* $(p, 2, \alpha)$*-sûr lorsque* $q_w < 1/8 \quad et \quad p_w < 1/8$.

Preuve Dans le cas d'un schéma à deux utilisateurs, une coalition contient soit un unique utilisateur, soit tous les utilisateurs du système. Dans le second cas, la procédure de traçage a systématiquement raison, puisqu'elle renvoie toujours un ensemble non-vide, qui est nécessairement contenu dans $\{1, 2\}$. On considère donc exclusivement le cas d'un unique traître.

Soit $\mathcal{G}_0$ le jeu de traçage pour le schéma décrit en définition 4.9 :
- L'environnement Ω génère deux paires de clés (ek_1, dk_1) et (ek_2, dk_2) pour le schéma de chiffrement $(\mathsf{I}, \mathsf{E}, \mathsf{D})$.
- Le décodeur pirate $\mathcal{P}$ choisit $u \in \{1, 2\}$, envoie u à Ω et reçoit dk_u de Ω.
- Ω choisit un message m, calcule $(m_1, m_2) = \mathsf{G}(m)$, choisit aléatoirement une permutation σ sur $\{1, 2\}$, et envoie $(E_{ek_u}(m_{\sigma(u)}), E_{ek_{\bar{u}}}(m_{\sigma(\bar{u})}))$ à $\mathcal{P}$. Le décodeur pirate $\mathcal{P}$ répond avec m'. Si $\mathsf{A}(m, m')$ est vrai, Ω calcule $u' = \sigma(\mathsf{D}(m', m, m_1, m_2))$. L'environnement Ω fait autant de requêtes que nécessaire, jusqu'à ce qu'il ait calculé au moins N_p fois la valeur u'.

Le décodeur pirate $\mathcal{P}$ gagne le jeu $\mathcal{G}_0$ si la valeur la plus fréquemment trouvée pour u' par Ω est $\bar{u}$.

On considère le jeu $\mathcal{G}_1$, identique au jeu $\mathcal{G}_0$, à la différence que l'environnement envoie $(E_{ek_u}(m_{\sigma(u)}), E_{ek_{\bar{u}}}(m_{\sigma(u)}))$ au lieu de $(E_{ek_u}(m_u), E_{ek_{\bar{u}}}(m_{\sigma(\bar{u})}))$. On construit un adversaire $\mathcal{A}$ de la sécurité LoR-CPA du schéma de chiffrement à partir du décodeur pirate $\mathcal{P}$:
- $\mathcal{A}$ reçoit ek, choisit aléatoirement $b \in \{1, 2\}$, calcule une paire de clés (ek_b, dk_b) pour le schéma de chiffrement et définit $ek_{\bar{b}} = ek$.
- $\mathcal{P}$ choisit $u \in \{1, 2\}$ et transmets u à $\mathcal{A}$: si $u \neq b$, alors $\mathcal{A}$ s'arrête en donnant pour réponse un élément choisi aléatoirement dans $\{1, 2\}$. Sinon, $\mathcal{A}$ envoie dk_u à $\mathcal{P}$ et le jeu se poursuit.
- $\mathcal{A}$ choisit un message m, calcule $(m_1, m_2) = \mathsf{G}(m)$, choisit aléatoirement une permutation σ sur $\{1, 2\}$, et envoie la requête $(m_{\sigma(1)}, m_{\sigma(2)})$ à son environnement. $\mathcal{A}$ reçoit la réponse $\mathsf{E}_{ek}(m_{\sigma(v)})$, où v est fixé. $\mathcal{A}$ envoie alors $(E_{ek_u}(m_{\sigma(u)}), E_{ek}(m_{\sigma(v)}))$ à $\mathcal{P}$, qui répond éventuellement avec un message m' : si $\mathsf{A}(m, m') = vrai$, $\mathcal{A}$ calcule $u' = \sigma(\mathsf{D}(m, m_1, m_2, m'))$. $\mathcal{A}$ réalise cette étape en boucle jusqu'à obtenir N_p résultats pour u'.

– $\mathcal{A}$ s'arrête en donnant pour réponse v', la valeur la plus souvent trouvée pour u' dans $\{1,2\}$.

Le schéma de chiffrement étant adv_e-LoR-CPA sûr, l'avantage de l'adversaire $\mathcal{A}$ construit précédemment est borné par adv_e. On en déduit que la différence des avantages du décodeur pirate $\mathcal{P}$ dans les jeux $\mathcal{G}_0$ et $\mathcal{G}_1$ est bornée par $2adv_e$:

$$\begin{aligned} adv_e &\geqslant |\Pr[v' = \bar{u} \,/\, (v = \bar{u} \text{ et } u = b)] - \Pr[v' = \bar{u} \,/\, (v = u \text{ et } u = b)]\,|\,/\,2\,, \\ &\geqslant |\Pr[v' = \bar{u} \text{ dans le jeu } \mathcal{G}_0] - \Pr[v' = \bar{u} \text{ dans le jeu } \mathcal{G}_1]\,|\,/\,2\,, \\ &\geqslant |\Pr[\mathcal{P} \text{ gagne dans le jeu } \mathcal{G}_0] - \Pr[\mathcal{P} \text{ gagne dans le jeu } \mathcal{G}_1]\,|\,/\,2\,. \end{aligned}$$

On considère désormais le jeu $\mathcal{G}_w$ entre un environnement Ω_w, et un adversaire $\mathcal{A}_w$ du schéma de marquage $(\mathsf{A}, \mathsf{G}, \mathsf{D})$:

– L'adversaire $\mathcal{A}_w$ choisit $u \in \{1,2\}$,
– L'environnement Ω_w choisit un message m, calcule $(m_1, m_2) = \mathsf{G}(m)$. Ω_w choisit aléatoirement une permutation σ sur $\{1,2\}$, et transmet $m_{\sigma(u)}$ à $\mathcal{A}_w$. L'adversaire $\mathcal{A}_w$ répond éventuellement avec m' : si $\mathsf{A}(m, m') = \mathit{vrai}$, Ω_w calcule $u' = \sigma(\mathsf{D}(m, m_1, m_2, m'))$. Ω_w réalise cette étape en boucle jusqu'à obtenir N_p résultats pour u'.

L'adversaire $\mathcal{A}_w$ gagne ce jeu si la valeur la plus fréquemment trouvée pour u' dans $\{1,2\}$ est $\bar{u}$. On remarque que l'adversaire $\mathcal{A}_w$ peut simuler des requêtes, via sa connaissance des algorithmes A, G et D. Ainsi, lors de chaque requête, l'adversaire $\mathcal{A}_w$ est dans une situation similaire à la première requête. Or le jeu sur la première requête est exactement le jeu de détection pour le schéma de marquage $(\mathsf{A}, \mathsf{G}, \mathsf{D})$.

Dans ce jeu $\mathcal{G}_w$, soit $N_p(0)$ le nombre de réponses données par $\mathcal{A}_w$ ayant conduit à $u' = 0$, soit $N_p(\bar{u})$ le nombre de réponses données par $\mathcal{A}_w$ ayant conduit à $u' = \bar{u}$. Concrètement, $N_p(0)$ est le nombre de fois où il y a échec dans la détection, tandis que $N_p(\bar{u})$ est le nombre de fois où il y a mauvaise détection. Lorsque l'adversaire $\mathcal{A}_w$ gagne le jeu $\mathcal{G}_w$, $N_p(u) \geqslant N_p(\bar{u})$, et on a nécessairement $N_p(\bar{u}) \geqslant N_p/3$ ou $N_p(0) \geqslant N_p/3$. Comme les requêtes se traduisent par des jeux indépendants (d'après le paragraphe précédent), on peut borner la probabilité de succès de $\mathcal{A}_w$:

$$\begin{aligned} \Pr[\mathcal{A}_w \text{ gagne le jeu } \mathcal{G}_w] &\leqslant \Pr[N_p(0) \geqslant N_p/3] + \Pr[N_p(\bar{u}) \geqslant N_p/3]\,, \\ &\leqslant \sum_{i=\lceil N_p/3 \rceil}^{N_p} C_{N_p}^i \, {q_w}^i + \sum_{i=\lceil N_p/3 \rceil}^{N_p} C_{N_p}^i \, {p_w}^i\,, \\ &\leqslant 2^{N_p} \, {q_w}^{N_p/3} + 2^{N_p} \, {p_w}^{N_p/3}\,, \\ &\leqslant (8\,q_w)^{N_p/3} + (8\,p_w)^{N_p/3}\,. \end{aligned}$$

Comme $p_w < 1/8$, $q_w < 1/8$, et grâce au choix adapté de la valeur de N_p, on en déduit que : $\Pr[\mathcal{A}_w \text{ gagne le jeu } \mathcal{G}_w] \leqslant p - 2adv_e$.

Il ne reste plus qu'à corréler les jeux $\mathcal{G}_1$ et $\mathcal{G}_w$ ce qui est assez aisé : à partir d'un décodeur pirate dans le jeu $\mathcal{G}_1$, on peut construire un adversaire $\mathcal{A}_w$ dans le jeu $\mathcal{G}_w$ avec exactement la même probabilité de succès. En effet, on peut considérer un adversaire

$\mathcal{A}_w$ qui génère des couples de clés, qui se contente de chiffrer à deux reprises la variante proposée par Ω_w, et de transmettre les résultats du chiffrement au décodeur pirate $\mathcal{P}$. Les probabilités de succès de $\mathcal{A}_w$ dans le jeu $\mathcal{G}_w$ et de $\mathcal{P}$ dans le jeu $\mathcal{G}_1$ sont identiques. On en déduit : $\Pr[\mathcal{P} \text{ gagne dans le jeu } \mathcal{G}_1] \leqslant p - 2adv_e$.

Le lien entre les probabilités de succès de $\mathcal{P}$ dans les jeux $\mathcal{G}_0$ et $\mathcal{G}_1$ permet d'obtenir le résultat attendu : $\Pr[\mathcal{P} \text{ gagne le jeu } \mathcal{G}_0] \leqslant 2adv_e + (p - 2adv_e) = p$. □

4.3.4 Propriétés du schéma

Le schéma élémentaire de traçage de traîtres présenté en définition 4.9 s'appuie sur un schéma de marquage et un schéma de chiffrement. L'exigence sur le schéma de marquage ($p_w < 1/8$ et $q_w < 1/8$) semble raisonnable à réaliser.

Il est difficile de parler de performances pour un schéma à seulement deux utilisateurs. Cependant, on peut évoquer le taux de chiffrement du schéma de traçage de traîtres : il est égal au double de celui du schéma de chiffrement. Dans un cadre symétrique, avec un algorithme bien choisi, on peut espérer avoir un taux de chiffrement proche de 1 pour le schéma de chiffrement, et donc un taux de chiffrement proche de 2 pour le schéma de traçage de traîtres.

En ce qui concerne le coût en temps de la procédure de traçage, elle nécessite un nombre de requêtes linéaire en $1/\alpha$, logarithmique en $p/2 - adv_e$. La probabilité d'erreur dans le traçage de traîtres est en général loin d'atteindre les niveaux de sécurité demandés dans les schémas de chiffrement classiques. On peut donc considérer que $p/2 - adv_e \approx p/2$. Le nombre de requêtes à réaliser est donc plutôt limité, et le traçage est ainsi plutôt efficace.

En tant que mécanisme de chiffrement, la sécurité du schéma élémentaire du traçage de traîtres est étroitement liée à la sécurité du schéma de chiffrement $(\mathsf{I}, \mathsf{E}, \mathsf{D})$. Il hérite donc d'une sécurité de type LoR-CPA.

On a déjà remarqué que l'utilisation d'un schéma de marquage empêche d'opter pour une méthode de chiffrement hybride : des clés de chiffrement de données ne peuvent pas être marquées. Par contre, si les données elles-mêmes peuvent être marquées, on peut marquer ces données de deux manières différentes, et chiffrer chacune de ces variantes par une clé générée aléatoirement. Ainsi, le chiffrement d'un message m devient

$$E_{ek}(m) = (\mathsf{E}_{ek_1}(sk_1), \mathsf{E}_{ek_2}(sk_2), \mathsf{S}_{sk_1}(m_{\sigma(1)}), \mathsf{S}_{sk_2}(m_{\sigma(2)})),$$

où S est un schéma de chiffrement symétrique rapide, et où sk_1 et sk_2 sont des clés secrètes choisies aléatoirement pour ce schéma symétrique. Les performances en termes de taux de chiffrement restent similaires, mais le temps de déchiffrement est amélioré.

On a traité le cas du schéma élémentaire de traçage de traîtres sûr dans un modèle fort, et efficace dans un cadre à deux utilisateurs. Il faut maintenant s'intéresser aux moyens de combiner différentes instances de ce schéma pour obtenir un schéma à n utilisateurs.

4.4 Codes résistants à des coalitions

On dispose désormais d'un schéma de traçage de traîtres pour deux utilisateurs. Sur un nombre plus important d'utilisateurs, cela signifie que si on distribue une clé de déchiffrement à la moitié des utilisateurs, et l'autre clé de déchiffrement à la seconde moitié, on est en mesure d'identifier une moitié d'utilisateurs contenant au moins un traître.

On souhaite désormais pouvoir recouper plusieurs informations de ce type afin de restreindre la taille de l'ensemble contenant un traître, et ceci jusqu'à identifier précisément un traître. L'idée est alors d'utiliser plusieurs instances du schéma de traçage de traîtres à deux utilisateurs, chacun d'entre eux permettant d'obtenir une information partielle : l'ensemble de ces informations doit permettre d'identifier un traître.

Ce principe simple, présenté dans [KY02], pose en fait le problème complexe du recoupement des informations partielles obtenues par chacune des instances du schéma de traçage de traîtres à deux utilisateurs. Le rapprochement avec la problématique des codes résistants à des coalitions, présentée dans [BS95], permet cependant de construire des solutions efficaces.

4.4.1 Traître unique ou plusieurs traîtres ?

En fait, dans le cas d'un unique traître, le problème est assez simple à résoudre. À l'inverse, dès que la coalition contient plusieurs traîtres, la situation est beaucoup plus complexe.

4.4.1.1 Traître unique

Dans le cas de coalitions limitées à un unique traître, chaque information obtenue indique un ensemble dont le traître fait partie. Ainsi, l'intersection de ces ensembles successifs contient le traître recherché (voir figure 4.1). Pour n utilisateurs, il suffit alors de $\lceil \log_2(n) \rceil$ informations partielles pour identifier avec certitude le traître.

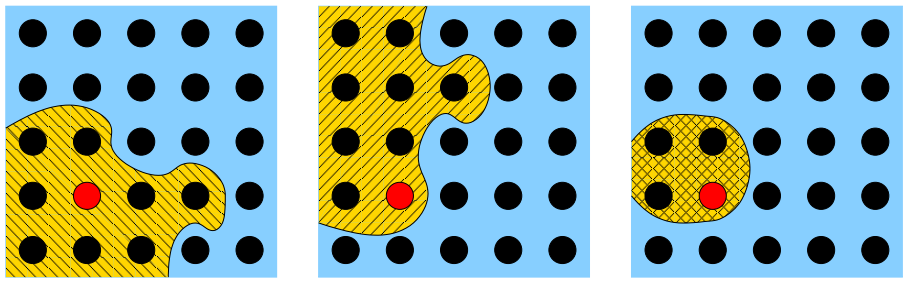

FIG. 4.1 – Résultats de deux instances et leur intersection pour un traître.

4.4.1.2 Deux traîtres ou plus

Dans le cas de coalitions de deux traîtres ou plus, chaque information obtenue indique un ensemble dont l'un des traîtres fait partie. L'intersection des ensembles déterminés n'est donc plus envisageable, deux ensembles pouvant contenir des traîtres différents (voir figure 4.2). Le recoupement des informations est donc plus délicat, dès lors que les coalitions peuvent regrouper plusieurs traîtres. On peut même prouver que face à une coalition contenant au moins trois traîtres, il est impossible d'identifier un traître avec certitude par le biais de codes binaires.

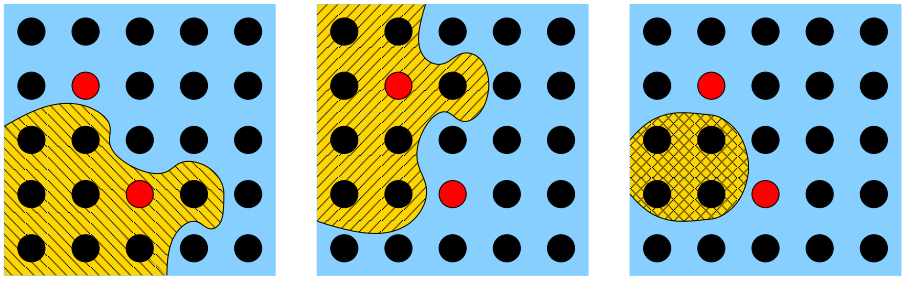

FIG. 4.2 – Résultats de deux instances et leur intersection pour deux traîtres.

4.4.2 Formalisme et codes sûrs contre des coalitions

4.4.2.1 Formalisme d'une instance

Lors de l'utilisation d'une instance d'un schéma de traçage de traîtres à deux utilisateurs, deux clés de déchiffrement distinctes et identifiables sont générées. Dans une utilisation pour n utilisateurs, on attribue à chacun des utilisateurs l'une des deux clés de déchiffrement : on doit définir précisément quelle clé reçoit chacun de ces utilisateurs. On assimile les utilisateurs aux entiers compris entre 1 et n. Pour un utilisateur $u \in \{1, \ldots, n\}$, soit ϕ_u un bit correspondant à la clé de déchiffrement reçu par l'utilisateur u (0 pour la première clé, 1 pour la seconde clé).

Des messages ou parties de messages sont envoyés protégés par le chiffrement défini par le schéma de traçage de traîtres. Un utilisateur honnête utilise sa clé de déchiffrement pour obtenir ces messages en clair. Un décodeur pirate construit par une coalition C dispose potentiellement de l'ensemble des clés de déchiffrement attribuées aux traîtres. Deux cas se présentent :

- tous les traîtres ont reçu la même clé de déchiffrement, et le décodeur pirate n'a pas d'autre choix que d'utiliser cette clé pour déchiffrer les données,
- deux traîtres ont reçu deux clés de déchiffrement distinctes, et le décodeur pirate peut utiliser celle de son choix, voire même les deux clés simultanément ou alternativement.

Dans le premier cas, le décodeur pirate se comporte de manière analogue à un décodeur pirate dans le schéma à deux utilisateurs, avec une connaissance limitée à la

clé de déchiffrement de l'un des deux utilisateurs uniquement. Le traçage pour deux utilisateurs permet d'identifier la clé de déchiffrement utilisée.

Dans le second cas, quelle que soit la réponse du traçage pour le schéma à deux utilisateurs, elle est valide puisque le décodeur pirate a eu accès aux deux clés de déchiffrement.

Globalement, dans les deux cas, si la procédure de traçage réussit, elle identifie une clé de déchiffrement détenue par la coalition, c'est-à-dire associée à un bit ϕ vérifiant :

$$\exists u \in C \,/\, \phi_u = \phi.$$

4.4.2.2 Formalisme de plusieurs instances

Lors de l'utilisation de l instances d'un schéma de traçage de traîtres à deux utilisateurs, on génère deux clés de déchiffrement distinctes et identifiables pour chaque instance. Chacun des n utilisateurs reçoit pour chacune des l instances l'une des deux clés de déchiffrement. L'attribution de ces clés à l'utilisateur $u \in \{1, \ldots, n\}$ est définie par un mot $w^{(u)}$ de l bits : le $i^{\text{ème}}$ bit $w_i^{(u)}$ de ce mot correspond comme précédemment à la clé de déchiffrement reçue par l'utilisateur u pour l'instance i (0 pour la première clé, 1 pour la seconde clé).

Un message envoyé est divisé en l parties complémentaires, chacune d'entre elles étant dans un premier temps supposée indispensable pour retrouver la moindre information sur le message envoyé. Chaque partie est alors chiffrée en utilisant une instance différente du schéma de chiffrement. On suppose que le traçage de chaque instance peut être effectué indépendamment des autres instances.

Dans ce contexte, pour chacune des l instances, le comportement d'une coalition C est similaire au comportement de la même coalition face à une instance unique. Ainsi, lorsque le traçage global réussit, il permet d'obtenir un mot w de l bit vérifiant la propriété fondamentale suivante : $\forall i \in \{1, \ldots, l\}$, $\exists u \in C \,/\, w_i = w_i^{(u)}$.

4.4.2.3 Codes binaires et coalitions

La propriété fondamentale présentée précédemment est très proche de celle des codes sûrs contre des coalitions. La répartition des clés de déchiffrement des l instances entre les n utilisateurs peut donc s'appuyer sur de tels codes.

Les codes sûrs contre des coalitions ont été introduits dans [BS95]. L'objectif est de construire différentes version d'un même document numérique, afin de pouvoir identifier la source d'une divulgation de ce document. L'idée est d'insérer dans le contenu numérique des bits, appelés marques, n'apportant aucune information nouvelle mais différents d'une copie à une autre.

Le détenteur d'une seule copie n'est pas en mesure de distinguer le contenu réel des marques qui ont été insérées, et il ne peut donc supprimer les marques. Une coalition dispose, elle, de plusieurs copies différentes du même contenu. Elle peut donc identifier les marques pour lesquelles différentes valeurs ont été utilisées dans les copies qu'elle

détient : les marques en question peuvent être détruites, ou prendre une valeur au choix de la coalition. En revanche, les marques pour lesquelles toutes les copies de la coalition sont identiques ne peuvent être distinguées du contenu et ne peuvent être changées.

Les codes sûrs contre des coalitions définissent une répartition des marques à utiliser entre les différentes copies distribuées afin de pouvoir identifier un membre d'une coalition, à partir du contenu sous la forme divulguée par cette coalition.

Un (l, n)-code binaire est une suite de n éléments de $\{0,1\}^l : (w^{(u)})_{1 \leqslant u \leqslant n} \in \left(\{0,1\}^l\right)^n$. Pour tout $u \in \{1, \ldots, n\}$, le mot de code $w^{(u)}$ correspond aux marques placées dans la copie distribuée à l'utilisateur u. Une coalition de t utilisateurs est représentée par un sous-ensemble C de $\{1, \ldots, n\}$. Pour tout $i \in \{1, \ldots, l\}$, une telle coalition peut identifier et altérer la marque en position i lorsque deux membres de cette coalition ont reçu des valeurs différentes de cette marque. La coalition C peut ainsi construire une copie avec les marques associées au mot $w \in \{0, 1, ?\}^l$ vérifiant la propriété suivante (le signe ? signifie que la marque a été supprimée ou altérée) :

$$\forall i \in \{1, \ldots, l\}\,, \begin{cases} w_i \in \{0,1\} \Longrightarrow \exists u \in C \,/\, w_i^{(u)} = w_i\,, \\ w_i = ? \Longrightarrow \exists (u, u') \in C^2 \,/\, w_i^{(u)} \neq w_i^{(u')}\,. \end{cases}$$

On appelle $F(C)$ l'ensemble des mots w vérifiant la propriété précédente, c'est-à-dire correspondant aux copies réalisables par la coalition C. On remarque en particulier que lors de l'utilisation de plusieurs instances d'un schéma de traçage de traîtres à deux utilisateurs, le mot w issu du traçage des instances respecte des contraintes de construction similaires, à partir des mots $w^{(u)}$ correspondant aux clés de déchiffrement des membres de la coalition.

4.4.2.4 Codes sûrs contre des coalitions

Idéalement, on souhaiterait pouvoir construire un code pour lequel la connaissance de $w \in F(C)$ permette de remonter de manière certaine à un membre de la coalition C. Il existe cependant un résultat fort (donné dans [BS95]) montrant l'impossibilité d'une telle construction. On considère en effet une coalition C de trois utilisateurs, u_1, u_2 et u_3. Cette coalition peut, pour toute marque détectable, prendre la marque majoritaire parmi ses utilisateurs :

$$\forall i \in \{1, \ldots, l\},\ w_i = \textit{Majorité}\left((w_i^{(u)})_{u \in C}\right).$$

Les marques w ainsi définies pourraient avoir été construites par deux membres quelconques de la coalition, sans l'aide du troisième :

$$w \in F(\{u_1, u_2\}), \quad w \in F(\{u_1, u_3\}), \quad w \in F(\{u_2, u_3\}).$$

Il n'est donc pas possible d'incriminer un utilisateur avec certitude, puisque la contribution d'un membre de la coalition n'est pas rigoureusement indispensable.

Cette faiblesse est d'autant plus pénalisante que les marques sont restreintes à des bits. En effet, en utilisant des symboles plus nombreux pour les marques, il faut une taille de coalition plus importante pour pouvoir mettre en place cette attaque. Cependant, dès que la taille de la coalition est supérieure à celle de l'ensemble des symboles utilisés comme marques, la faiblesse est bien présente. Pour pouvoir espérer tracer avec certitude des coalitions de taille t, il faut donc au moins t symboles pour les marques. Cette contrainte peut paraître raisonnable, mais dans le contexte du traçage de traîtres, cela induit nécessairement un schéma élémentaire à t utilisateurs, et non plus deux : comme l'utilisation du watermarking présentée précédemment induit nécessairement une taille des chiffrés linéaire en le nombre d'utilisateurs du schéma élémentaire, cette option n'est pas envisageable si on veut préserver une taille des chiffrés indépendante du nombre t de traîtres regroupés dans la coalition.

Pour préserver des marques binaires, l'alternative consiste en un traçage probabiliste des marques. Pour cela, on ne s'intéresse pas à un code binaire unique mais à une famille de codes : on génère un code de la famille en suivant une procédure probabiliste. La robustesse du code vient du fait qu'une coalition ne sait pas précisément quel code a été généré.

Plus concrètement, un code binaire $(w^{(u)})_{1\leqslant u\leqslant n}$ est tiré aléatoirement dans une famille $\mathcal{F}$ de codes lors du marquage du contenu à protéger. On distribue les n versions marquées, chacune d'entre elles étant associée à un mot du code $w^{(u)}$. La connaissance de la famille $\mathcal{F}$ est publique, mais le code binaire utilisé est gardé secret.

Une coalition C d'au plus t utilisateurs reçoit autant de contenus qu'elle a de membres : elle connaît donc au mieux $(w^{(u)})_{u\in C}$, c'est-à-dire au plus t mots du code utilisé [2]. Tous les autres mots de code lui sont inconnus : la coalition est donc incapable d'identifier le code utilisé pour marquer les contenus dans l'ensemble de tous les codes $(x^{(u)})_{1\leqslant u\leqslant n}$ de la familles $\mathcal{F}$ vérifiant : $\forall u \in C, x^{(u)} = w^{(u)}$. Quels que soient ces choix, la coalition agit donc de manière similaire pour tous ces codes.

La définition suivante exige qu'il existe une procédure de traçage τ telle que quelles que soient les marques générées par la coalition, ce traçage permette d'identifier des membres de cette coalition avec une probabilité $1-p$ sur l'ensemble des codes $(x^{(u)})_{1\leqslant u\leqslant n}$ de $\mathcal{F}$ coïncidant avec le code $(w^{(u)})_{1\leqslant u\leqslant n}$ sur C, c'est-à-dire tels que $\forall u \in C, x^{(u)} = w^{(u)}$. Cela signifie que quel que soit le comportement de la coalition, elle peut au mieux masquer ses membres pour une fraction p de cas totalement indistinguables de son point de vue.

Définition 4.10 (Famille de codes binaires sûre contre des coalitions) *Une famille $\mathcal{F}$ de (l, n)-codes binaires est (p, t)-sûre contre des coalitions si on peut tracer des coalitions d'au plus t membres avec un probabilité d'erreur au plus p.*

Plus précisément, il existe une procédure de traçage τ prenant pour entrées la description d'un code et un mot dans $\{0, 1, ?\}^l$ associé à des marques, et calculant un ensemble d'utilisateurs tel que : pour tout code $(w^{(u)})_{1\leqslant u\leqslant n}$ de la famille $\mathcal{F}$, pour toute

[2] Si certaines marques sont identiques dans toutes les versions distribuées aux membres de la coalition, elle n'est en fait pas capable d'identifier complètement les mots de code associés.

coalition C de taille inférieure ou égale à t, pour tout mot $w \in F(C)$, la probabilité sur l'ensemble des codes $(x^{(u)})_{1\leqslant u\leqslant n}$ de $\mathcal{F}$ coïncidant avec $(w^{(u)})_{1\leqslant u\leqslant n}$ sur C que la procédure τ appliquée au code $(x^{(u)})_{1\leqslant u\leqslant n}$ et au mot w retourne un ensemble d'utilisateurs vide ou non-inclus dans C est inférieure à p.

Cette définition n'a de sens que pour une famille de codes. Cependant, lorsque la description du code induit implicitement la famille de codes qui correspond, parler simplement de code binaire sûr contre des coalitions est envisageable, et c'est d'ailleurs le cas dans [BS95] où la notion de famille est masquée.

4.4.3 Effacements : notion et formalisme

4.4.3.1 Des effacements presque incontournables

Comme on l'a vu précédemment, il y a une forte adéquation entre le choix d'une répartition de clés de déchiffrement de plusieurs instances pour le traçage de traîtres et les codes sûrs contre les coalitions. Cependant, l'utilisation d'une famille de codes sûre contre des coalitions dans un schéma de traçage de traîtres, telle que présentée en paragraphe 4.4.2.2, se heurte dans notre modèle à une contradiction concernant l'indépendance des instances :

- d'une part, les instances doivent être corrélées pour empêcher un décodeur pirate de se passer des clés de déchiffrement correspondant à certaines instances ;
- d'autre part, le traçage individuel de chacune des instances nécessite a priori que les instances soient indépendantes.

Pour expliciter cette contradiction, on suit dans un premier temps l'approche donnée dans [KY02]. La construction initiale prévoit l'utilisation d'instances indépendantes d'un schéma de traçage de traîtres élémentaire (c'est-à-dire à deux utilisateurs). Ainsi, un message est découpé en parties indépendantes, chacune d'entre elles étant chiffrée par une instance différente. Dans ce contexte, un décodeur pirate peut ne déchiffrer qu'une partie du message, en ne déchiffrant pas les parties correspondant à une ou plusieurs instances. Un tel décodeur pirate n'obtient qu'une partie des données, mais empêche le traçage des instances pour lesquelles il refuse de déchiffrer. Ce faisant, il empêche le traçage global, certaines position du mot de code ne pouvant être déterminées par le traceur.

Pour forcer l'utilisation systématique d'une clé de déchiffrement pour chacune des instances du schéma élémentaire, il est proposé dans [KY02] de créer une dépendance entre les messages chiffrés par chacune des instances. En pratique, on utilise une transformation « tout ou rien » (*All-or-Nothing transform*) construite dans [Riv97]. Le principe de ce mécanisme est de corréler $l-1$ blocs de contenu, sous la forme de l blocs de même taille. Tant qu'un de ces l blocs est inconnu, on ne peut identifier aucun des $l-1$ blocs du contenu initial. Ainsi, en chiffrant chacun de ces l blocs par une instance d'un schéma de traçage de traîtres, si un décodeur pirate renonce à déchiffrer une instance particulière, il se prive de l'ensemble des informations contenues dans toutes les instances, ce qui est inacceptable.

En termes de performances, cette méthode présente l'avantage de ne dégrader que très faiblement le rapport entre la taille du contenu et la taille du chiffré transmis. Par contre, elle induit des délais : un décodeur doit attendre d'avoir obtenu un ensemble de l blocs avant de pouvoir obtenir des données.

Cette méthode empêche donc un décodeur pirate de refuser de déchiffrer certaines instances du schéma élémentaire de traçage de traîtres. Mais comme les instances ne sont plus indépendantes, la possibilité de tracer individuellement les instances peut du coup disparaître.

C'est typiquement le cas avec les techniques de watermarking visant à rendre les messages de traçage indistinguables des messages de contenu, et donc à assurer un traçage effectif des instances dans un modèle fort. En effet, avec la transformation « tout ou rien », de légères modifications sur un bloc de données ont des répercussions importantes sur l'ensemble de tous les blocs. Ainsi, l'utilisation de deux versions différentes d'un bloc de données n'est possible que si l'on crée pour chacune de ces deux versions un ensemble de blocs cohérents, et qu'on s'assure qu'un destinataire d'une version donnée du bloc ciblé reçoit l'ensemble des blocs cohérents de cette version. Comme la répartition des clés de déchiffrement est différente d'une instance à une autre, on ne peut pas satisfaire cette contrainte. Cette méthode présente donc des incompatibilités fortes avec certains schémas élémentaires de traçage de traîtres, et c'est en fait systématique pour les schémas élémentaires sûrs dans le modèle fort présenté en section 4.2.3.

On peut renoncer à un modèle fort pour le traçage de traîtres. Dans ce cas, l'interdépendance des instances peut rester compatible avec le schéma de traçage de traîtres élémentaire utilisé. Mais le fait que les blocs de contenu soient fortement corrélés les uns aux autres rend le traçage beaucoup plus coûteux en temps. En effet, sans la transformation « tout ou rien », on peut tracer simultanément toutes les instances, et envoyer un message de traçage composé de l blocs de traçage pour le schéma élémentaire. Avec la transformation « tout ou rien », la seule méthode envisageable est d'envoyer des messages de traçage composés de $l-1$ blocs valides, et d'un bloc de traçage pour une seule instance. On multiplie donc par l le temps nécessaire au traçage complet d'un décodeur pirate.

Ainsi, non seulement tout décodeur doit déchiffrer l blocs avant de pouvoir obtenir la moindre informations sur le message associé, mais le traçage complet d'un décodeur pirate nécessite un envoi de l^2 fois plus de données que le traçage du schéma élémentaire. Ces deux inconvénients sont d'autant plus gênants que la valeur de l est élevée, ce qui est en fait le cas. Par exemple, en utilisant les valeurs numériques données dans [CFN94] (probabilité d'erreur limitée à 2^{-10} pour des coalitions d'au plus 32 traîtres parmi 10^9 utilisateurs), le code de taille logarithmique proposé dans [BS95] aboutit à $l \approx 5.10^8$ instances.

Pour rester dans un modèle de sécurité fort, ou simplement pour éviter ces inconvénients, il faut opter pour l'approche inverse : au lieu de corréler les instances pour assurer une utilisation systématique d'une clé de déchiffrement par instance, on doit accepter que le décodeur pirate puisse ne pas répondre à certaines instances du schéma

élémentaire de traçage de traîtres. Cela se traduit par des effacement sur le mot obtenu, suite au traçage des instances. La famille de codes doit ainsi tolérer des effacements.

4.4.3.2 Définition des codes tolérant des effacements

On a vu précédemment (partie 4.2) qu'il était raisonnable pour un décodeur pirate de ne pas déchiffrer systématiquement tous les messages reçus, mais qu'il devait déchiffrer avec un taux minimum pour pouvoir être utilisable. En termes de codes, cela signifie que certaines positions peuvent être effacées, mais en proportion limitée.

Des familles de codes sûres contre des coalitions et tolérant des effacements ont déjà fait l'objet d'études approfondies. Ainsi, dans [GP99], une définition et une construction est proposée pour supporter de tels effacements dans le cas binaire. Dans [SNW01], la définition précédente est étendue au cas non-binaire, et une nouvelle construction est proposée. Ces constructions ne sont cependant pas adaptées à notre contexte, puisqu'elles supposent que les effacements sont répartis uniformément dans l'ensemble des positions du code, ce qui correspond au contexte d'une dégradation du message lors de sa transmission. Pour l'application au traçage de traîtres, la répartition des effacements n'est pas uniformément répartie. Elle est au contraire au libre choix du décodeur pirate, et elle peut notamment dépendre des clés de déchiffrement distribuées aux membres de la coalition : on ne peut donc pas supposer que cette répartition soit uniformément répartie sur l'ensemble des positions.

Les codes pour des marques réduites (*codes for shortened fingerprints*) présentés dans [SNW01, SNW02] ont quant à eux des exigences plus fortes : dans leur contexte, la position des effacements est inconnue, et le mot caractéristique des marques a une longueur plus petite que celle des mots du code. Les codes construits pour ces définitions pourraient être utilisés dans le cadre du traçage de traîtres, mais ils sont basés sur des alphabets de taille importante : comme le rapport de taille entre les messages chiffrés et les messages clairs est linéaire en la taille de l'alphabet du code dans le schéma de traçage de traîtres élémentaire, utiliser ces codes n'est pas envisageable.

On doit donc construire des codes binaires tolérant des effacements. Les définitions suivantes correspondent à une version binaire de la définition donnée dans [SNW01], et à une version sans hypothèse de répartition uniforme des effacements de la définition donnée dans [GP99] :

Définition 4.11 (Ensemble de mots réalisables avec effacements) *Pour tout (l,n)-code binaire $(w^{(u)})_{1\leqslant u\leqslant n}$, on note $F_\alpha(C)$ l'ensemble des mots $w \in \{0,1,\perp\}^l$ associés aux marques réalisables par une coalition $C \subset \{1,\dots,n\}$ avec un taux de réponse minimum α, c'est-à-dire l'ensemble des mots $w \in \{0,1,\perp\}^l$ vérifiant les deux propriétés suivantes ($\perp$ indique un effacement) :*

$$\begin{cases} \forall i \in \{1,\dots,l\}\,, \left(w_i \in \{0,1\} \Longrightarrow \exists u \in C \,/\, w_i^{(u)} = w_i\right), \\ \#\left\{ i \in \{1,\dots,l\} \,/\, w_i \neq \perp \right\} \geqslant \alpha\, l\,. \end{cases}$$

Définition 4.12 (Codes binaires sûrs contre des coalitions avec effacements) *Une famille $\mathcal{F}$ de (l,n)-codes binaires est (p,t,α)-sûre contre des coalitions avec effacements si on peut tracer des coalitions d'au plus t membres avec un probabilité d'erreur au plus p, lorsque le taux de réponse est supérieur à α.*

Plus précisément, il existe une procédure de traçage τ prenant pour entrées la description d'un code et un mot dans $\{0,1,\perp\}^l$, et calculant un ensemble d'utilisateurs tel que : pour tout code $(w^{(u)})_{1\leqslant u\leqslant n}$ de la famille $\mathcal{F}$, pour toute coalition C de taille inférieure ou égale à t, pour tout mot $w \in F_\alpha(C)$, la probabilité sur l'ensemble des codes $(x^{(u)})_{1\leqslant u\leqslant n}$ de $\mathcal{F}$ coïncidant avec $(w^{(u)})_{1\leqslant u\leqslant n}$ sur C que la procédure τ appliquée au code $(x^{(u)})_{1\leqslant u\leqslant n}$ et au mot w retourne un ensemble d'utilisateurs vide ou non-inclus dans C est inférieure à p.

La définition précédente est naturellement plus forte que la définition 4.10 : pour tout α, une famille de (l,n)-codes binaires (p,t,α)-sûre contre des coalitions avec effacements est nécessairement (p,t)-sûre contre des coalitions.

4.4.4 Code $\Gamma_0(n,d)$: coalitions quelconques sans effacements

4.4.4.1 Définition

Cette première famille de codes, appelée $\Gamma_0(n,d)$, est due à [BS95]. Il s'agit d'une famille de codes binaires sûre contre des coalitions de taille quelconque. Par rapport à [BS95], l'originalité de cette présentation réside principalement dans la preuve du théorème, bien plus détaillée, qui permet une adaptation facile ultérieurement.

Soient n et d deux entiers non nuls. Soit $l = nd - d$. Un code de la famille $\Gamma_0(n,d)$ est associé à une partition de l'ensemble $\{1,\ldots,l\}$ en $(n-1)$ sous-ensembles de taille exactement d : $(\Pi_j)_{1\leqslant j\leqslant n-1}$. Ce code est alors défini par $(w^{(1)},\ldots,w^{(n)})$ vérifiant :

$$\forall u \in \{1,\ldots,n\}, \forall j \in \{1,\ldots,n-1\}, \forall i \in \Pi_j, w_i^{(u)} = \begin{cases} 0 & \text{si } j < u\,, \\ 1 & \text{si } j \geqslant u\,. \end{cases}$$

Une manière simple de visualiser ce code est de considérer une permutation π des positions dans tous les mots de code, telle que : $\forall j \in \{1,\ldots,n-1\}$, $\forall i \in \Pi_j$, $(j-1)\,d < \pi(i) \leqslant j\,d$. Un mot de code dont les positions sont permutées est constitué par des blocs de 1, suivis par des blocs de 0, comme le montre la figure 4.3, chaque bloc étant de longueur d.

4.4.4.2 Procédure de traçage

Ainsi, le mot de code $w^{(1)}$ est le seul à avoir des 1 aux positions i appartenant à Π_1. De même, le mot de code $w^{(n)}$ est le seul à avoir des 0 aux positions i appartenant à Π_{n-1}. En ce qui concerne les mots de code intermédiaires, pour tout $u \in \{2,\ldots,n-1\}$, le mot de code $w^{(u)}$ est le seul à avoir des valeurs différentes sur les positions i appartenant à Π_{u-1} d'une part, et Π_u d'autre part. Le traçage tire parti de ces propriétés.

u	Permuté de $w^{(u)}$					
1	$1\cdots1$	$1\cdots1$	$1\cdots1$	$\cdots\cdots$	$1\cdots1$	$1\cdots1$
2	$0\cdots0$	$1\cdots1$	$1\cdots1$	$\cdots\cdots$	$1\cdots1$	$1\cdots1$
3	$0\cdots0$	$0\cdots0$	$1\cdots1$	$\cdots\cdots$	$1\cdots1$	$1\cdots1$
$\vdots$						
$n-1$	$0\cdots0$	$0\cdots0$	$0\cdots0$	$\cdots\cdots$	$0\cdots0$	$1\cdots1$
n	$0\cdots0$	$0\cdots0$	$0\cdots0$	$\cdots\cdots$	$0\cdots0$	$0\cdots0$
	$\overleftrightarrow{\Pi_1}$	$\overleftrightarrow{\Pi_2}$	$\overleftrightarrow{\Pi_3}$		$\overleftrightarrow{\Pi_{n-2}}$	$\overleftrightarrow{\Pi_{n-1}}$

FIG. 4.3 – Code de la famille $\Gamma_0(n,d)$

Pour le code correspondant à la partition $(\Pi_j)_{1\leqslant j\leqslant n-1}$, pour une probabilité d'erreur p, la procédure de traçage τ_p appliquée au mot w consiste dans un premier temps à calculer $c(j) = \#\{i \in \Pi_j \,/\, w_i = 1\}$, pour tout $j \in \{1,\dots,n-1\}$. La réponse donnée par cette procédure contient les éléments suivants :

- 1, si $c(1) > 0$;
- n, si $c(n-1) < d$;
- $u \in \{2,\dots,n-1\}$, si $|c(u) - c(u-1)| > 2\sqrt{(c(u)+c(u-1))\log(n/p)}$.

Plus concrètement, l'idée derrière cette procédure de traçage est que les premier et dernier blocs permettent d'identifier sans doute possible le premier et le dernier mot du code. Si le mot reçu ne contient que des 0 sur le premier bloc, et que des 1 sur le dernier bloc, il y a nécessairement une évolution de la proportion de 1 lors du parcours des blocs, du premier au dernier. Comme deux blocs consécutifs ne sont distinguables qu'en connaissant un mot de code bien précis, le traçage détermine les blocs consécutifs présentant la différence de proportions de 1 la plus significative (d'une certaine manière), et donne le mot permettant de distinguer ces blocs comme réponse. La figure 4.4 illustre cette interprétation.

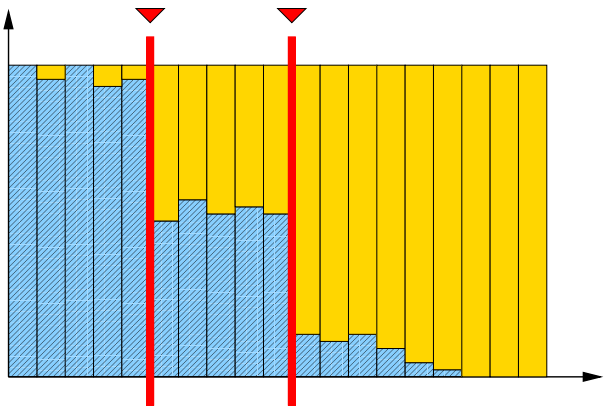

FIG. 4.4 – Représentation de $c(j)$ et traçage du mot associé.

4.4.4.3 Preuve de sécurité

Le théorème suivant montre que cette procédure de traçage est pertinente, pour toute coalition, lorsque l'entier d est suffisamment grand.

Théorème 4.2 *Pour n et p vérifiant $n \geqslant 8p$, la famille de $(nd-d, n)$-codes binaires de type $\Gamma_0(n,d)$, avec la procédure de traçage τ_p décrite précédemment, est (p,n)-sûre (sans effacements) contre des coalitions lorsque*

$$d > 2(n-1)(n-2)\log(n/p) \sim 2n^2\log(n/p).$$

Preuve La preuve consiste dans un premier temps à prouver que la réponse de la procédure de traçage contient un élément hors de la coalition avec une probabilité inférieure à p. Ensuite, on prouve que la réponse de la procédure de traçage est nécessairement un ensemble non-vide.

On considère un code binaire de type $\Gamma_0(n,d)$. Soit $(\Pi_j)_{1\leqslant j\leqslant n-1}$ la partition de l'ensemble $\{1,\ldots,nd-d\}$ caractérisant ce code. On considère une coalition $C = \{u_1, u_2, \ldots, u_t\} \subset \{1,\ldots,n\}$, où $u_1 < u_2 < \ldots < u_t$. Pour cette coalition, un code de type $\Gamma_0(n,d)$ est indistinguable du code utilisé si et seulement si sa partition $(\Pi'_j)_{1\leqslant j\leqslant n-1}$ de $\{1,\ldots,nd-d\}$ vérifie toutes les contraintes suivantes :

$$\bigcup_{1\leqslant j\leqslant u_1-1} \Pi_j = \bigcup_{1\leqslant j\leqslant u_1-1} \Pi'_j\,,$$

$$\bigcup_{u_t\leqslant j\leqslant n-1} \Pi_j = \bigcup_{u_t\leqslant j\leqslant n-1} \Pi'_j\,,$$

$$\forall k\in\{1,\ldots,t-1\},\quad \bigcup_{u_k\leqslant j\leqslant u_{k+1}-1} \Pi_j = \bigcup_{u_k\leqslant j\leqslant u_{k+1}-1} \Pi'_j.$$

Soit $u \in \{1,\ldots,n\}$ n'appartenant pas à C. Si $u = 1$, ou plus généralement si $u < u_1$, l'ensemble identifié par la procédure de traçage ne peut contenir u. En effet, tous les mots de la coalition C contiennent exclusivement des 1 pour les positions appartenant aux Π_j tels que $j < u_1$:

- si $u = 1$, $c(1) = 0$, et donc u n'est pas dans l'ensemble identifié ;
- si $u \neq 1$, $c(u) = c(u-1) = 0$, et donc u n'est pas dans l'ensemble identifié.

De la même manière, si $u = n$, ou plus généralement si $u > u_t$, la procédure de traçage ne peut identifier u comme appartenant à la coalition : $c(u) = c(u-1) = d$, et donc u n'est pas dans l'ensemble identifié.

Le seul cas restant est donc : $u \in \{u_1+1,\ldots,u_t-1\}$. Soit $k_0 \in \{1,\ldots,t\}$ tel que $u_{k_0} < u < u_{k_0+1}$. La répartition des codes indistinguables du code considéré engendre la probabilité suivante :

$$\forall i \in \bigcup_{u_{k_0}\leqslant j<u_{k_0+1}} \Pi'_j\,,\quad \forall j_0 \in \{u_{k_0},\ldots,u_{k_0+1}-1\}\,,\quad \Pr[i\in\Pi'_{j_0}] = \frac{1}{u_{k_0+1}-u_{k_0}}.$$

En particulier, en fixant successivement j_0 à $(u-1)$ puis à u, on obtient :

$$\forall i \in \bigcup_{u_{k_0} \leqslant j < u_{k_0+1}} \Pi'_j\,, \quad \Pr[i \in \Pi'_{u-1}] = \Pr[i \in \Pi'_u].$$

Soient $c'(u-1) = \#\{i \in \Pi'_{u-1} \,/\, w_i = 1\}$ et $c'(u) = \#\{i \in \Pi'_u \,/\, w_i = 1\}$. Le traçage dans le code associé à la partition $(\Pi'_j)_{1 \leqslant j \leqslant n-1}$ détecte u si et seulement si :

$$|c'(u) - c'(u-1)| > 2\sqrt{(c'(u) + c'(u-1))\log(n/p)}.$$

On doit donc étudier ces quantités $c'(u-1)$ et $c'(u)$ et plus précisément leur comportement relatif. Suite aux probabilités données précédemment, une position peut indifféremment appartenir à Π'_{u-1} ou à Π'_u. Ainsi,

$$\forall y \in \{0,\dots,2d\},\, \forall z \in \{0,\dots,y\}, \quad \Pr[c(u) = z \;/\; c(u) + c(u-1) = y] = \frac{C_d^z \cdot C_d^{y-z}}{C_{2d}^y}.$$

On note $f(d,y,z)$ la probabilité précédente, et on étudie le lien entre les quantités $f(d,y,z+1)$ et $f(d,y,z)$ lorsque $z > y/2$:

$$\forall z > y/2\,, \quad f(d,y,z+1) = \frac{(d-z)(y-z)}{(z+1)(d-y+z+1)}\, f(d,y,z) \;\leqslant\; \frac{y-z}{z+1}\, f(d,y,z).$$

Pour tout $x \in [0, y/2]$, soit $I(d,x,y) = \Pr[|c'(u) - y/2| > x \;/\; c'(u) + c'(u-1) = y]$. Par définition de I et par symétrie de f, $I(d,x,y)$ est deux fois la somme des $f(d,y,z)$ tels que $z > x + y/2$. On obtient ainsi :

$$I(d,x+1,y) \;=\; 2 \sum_{z > x+y/2} f(d,y,z+1) \;\leqslant\; \frac{y/2 - x}{y/2 + x + 1}\, I(d,x,y).$$

Or pour tout $\varepsilon \in [0,1]$, on peut majorer $(1-\varepsilon)/(1+\varepsilon)$ par l'exponentielle $e^{-2\varepsilon}$. On peut donc majorer le facteur de l'inégalité précédente par $e^{-2\frac{2x+1}{y+1}}$. On en déduit :

$$\forall x \in [0, y/2], \quad I(d,x,y) \;\leqslant\; I(d, x - \lfloor x \rfloor, y)\, e^{\frac{-2\lfloor x \rfloor (2x - \lfloor x \rfloor)}{y+1}} \;\leqslant\; e^{\frac{-2(x^2-1)}{y+1}}.$$

Ainsi, lorsque $y \geqslant 2$ et $x \geqslant \sqrt{y \log(n/p)}$, on a $I(d,x,y) \leqslant p/n$.

De plus, par définition,

– pour tout $x \geqslant 0$, $I(d,x,0) = 0$;

– pour tout $x \geqslant \sqrt{\log(n/p)} > 1$, $I(d,x,1) = 0$.

On obtient donc globalement : $\forall y \in \{0,\dots,2d\}, \forall x \geqslant \sqrt{y \log(n/p)}, I(d,x,y) \leqslant p/n$. Ce qui s'interprète par :

$$\Pr\left[|c'(u) - c'(u-1)| > 2\sqrt{(c'(u) + c'(u-1))\log(n/p)}\right] \leqslant p/n.$$

La probabilité que l'ensemble donné par la procédure de traçage contienne u est donc majorée par p/n. En prenant en compte tous les mots n'appartenant pas à la

coalition, la probabilité que l'ensemble donné par la procédure de traçage ne soit pas inclus dans C est bornée par p. Il reste donc à prouver que la procédure de traçage répond toujours avec un ensemble non-vide.

On suppose que la réponse de la procédure de traçage ne contient pas les éléments 1, 2, ..., $n-1$. Cela signifie :

$$c(1) = 0\,, \quad \forall j \in \{2, \ldots, n-1\},\ |c(j) - c(j-1)| \leqslant 2\sqrt{(c(j) + c(j-1))\log(n/p)}.$$

Pour tout $j \in \{1, \ldots, n-1\}$, soit $g(j) = 1/4 + c(j)/(2\log(n/p))$. On obtient :

$$g(1) = 1/4\,, \quad \forall j \in \{2, \ldots, n-1\},\ (g(j) - g(j-1) - 1)^2 \leqslant 4g(j-1).$$

On en déduit : $\forall j \in \{2, \ldots, n-1\},\ \sqrt{g(j)} \leqslant \sqrt{g(j-1)} + 1$. D'où :

$$g(n-1) \leqslant (n - 3/2)^2\,, \text{ c'est-à-dire :} \quad c(n-1) \leqslant 2\,(n-1)(n-2)\log(n/p).$$

Lorsque $d > 2(n-1)(n-2)\log(n/p)$, $c(n-1) < d$, et la procédure de traçage répond avec le singleton $\{n\}$. La procédure de traçage ne peut donc pas répondre avec un ensemble vide, ce qui conclut la preuve. □

4.4.4.4 Performances

Les performances de la famille de codes $\Gamma_0(n, d)$ sont limitées : la longueur du code est polynomiale (cubique) en le nombre de mots du code. Par contre, cette famille prouve l'existence de familles de codes binaires sûres contre des coalitions de toute taille, pour toute probabilité d'erreur requise. De plus, la dépendance de la longueur du code en la probabilité d'erreur p est logarithmique, ce qui est plutôt attractif.

4.4.5 Code $\Gamma'_0(n, d, \alpha)$: coalitions quelconques avec effacements

4.4.5.1 Introduction

On généralise cette famille de codes, sous la forme de la famille de codes notée $\Gamma'_0(n, d, \alpha)$. La généralisation porte sur le taux de réponse nécessaire pour assurer le traçage : la famille $\Gamma'_0(n, d, \alpha)$ permet un traçage avec un taux de réponse supérieur ou égal à α, alors que la famille $\Gamma_0(n, d)$ nécessitait un taux de réponse égal à 1.

En fait, cette généralisation consiste en une modification de la procédure de traçage, et la sécurité nécessite un allongement de la valeur de d en fonction de n. Par contre, le principe de la famille de codes reste le même : utiliser des blocs successifs distinguables par un unique utilisateur.

Pour construire cette famille et comprendre la méthode de traçage, on revient sur la famille de codes $\Gamma_0(n, d)$: en quoi des effacements peuvent-ils perturber le traçage dans la famille $\Gamma_0(n, d)$? Si les effacements sont répartis quasi-uniformément sur tous les blocs, la méthode de traçage va fonctionner de manière similaire, mais avec une précision

Permuté de $w^{(1)}$	$1\cdots1$	$1\cdots1$	$1\cdots1$	$\cdots\cdots$	$1\cdots1$	$1\cdots1$
Permuté de $w^{(2)}$	$0\cdots0$	$1\cdots1$	$1\cdots1$	$\cdots\cdots$	$1\cdots1$	$1\cdots1$
Permuté de $w^{(3)}$	$0\cdots0$	$0\cdots0$	$1\cdots1$	$\cdots\cdots$	$1\cdots1$	$1\cdots1$
$\vdots$						
Permuté de $w^{(n)}$	$0\cdots0$	$0\cdots0$	$0\cdots0$	$\cdots\cdots$	$0\cdots0$	$0\cdots0$
Permuté de w	$0\cdots0$	$\perp\cdots\perp$	$1\cdots1$	$\cdots\cdots$	$1\cdots1$	$1\cdots1$
	$\overleftrightarrow{\Pi_1}$	$\overleftrightarrow{\Pi_2}$	$\overleftrightarrow{\Pi_3}$		$\overleftrightarrow{\Pi_{n-2}}$	$\overleftrightarrow{\Pi_{n-1}}$

FIG. 4.5 – Effacement de Π_2 sur un code de la famille $\Gamma_0(n,d)$

de traçage plus faible. Par contre, si les effacements sont concentrés sur certains blocs, on perd toute l'information sur ces blocs, et on ne peut plus tracer en utilisant ces blocs. Les figures 4.5 et 4.6 montrent ce phénomène : la « discontinuité » créée par les effacements bloque le traçage.

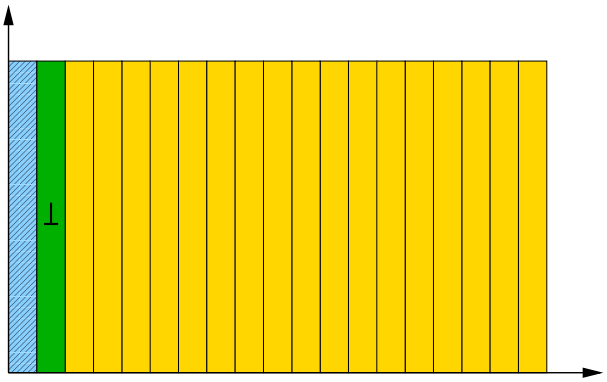

FIG. 4.6 – Représentation de $c(j)$ avec effacement de Π_2 : traçage impossible.

En revanche, l'indistinguabilité des blocs successifs reste vraie : si certains blocs sont presque totalement effacés, alors que des blocs voisins sont à peine touchés par les effacements, il est probable que le mot distinguant ces blocs appartienne à la coalition, et permette ainsi le choix d'effacements ciblés. Ceci nous conduit à une procédure de traçage fonctionnant de la manière suivante :

- si les effacements sont à peu près uniformément répartis sur les blocs, le traçage s'appuie comme précédemment sur la proportion de 0 et de 1 dans les blocs ;
- si en revanche certains blocs sont fortement effacés, alors le traçage s'appuie sur la proportion d'effacements dans les blocs.

4.4.5.2 Définition

Soient n et d deux entiers non nuls, soit $\alpha \in]0,1]$. Soit $l = nd - d$. Un code de la famille $\Gamma'_0(n,d,\alpha)$ est un code de la famille $\Gamma_0(n,d)$, c'est-à-dire qu'il est associé à une

partition de l'ensemble $\{1,\dots,l\}$ en $(n-1)$ sous-ensembles de taille exactement d : $(\Pi_j)_{1\leqslant j\leqslant n-1}$. Ce code est alors défini par $(w^{(1)},\dots,w^{(n)})$ vérifiant :

$$\forall u\in\{1,\dots,n\},\forall j\in\{1,\dots,n-1\},\forall i\in\Pi_j, w_i^{(u)}=\begin{cases}0 & \text{si } j<u\,,\\ 1 & \text{si } j\geqslant u\,.\end{cases}$$

4.4.5.3 Procédure de traçage

Pour le code de la famille $\Gamma_0'(n,d,\alpha)$ correspondant à la partition $(\Pi_j)_{1\leqslant j\leqslant n-1}$, la procédure de traçage utilise la probabilité d'erreur p, le taux de réponse minimum α, mais également un troisième paramètre $\beta\in]0,\alpha]$, multiple entier de $1/d$. La procédure de traçage $\tau'_{(p,\alpha,\beta)}$ appliquée au mot $w\in\{0,1,\perp\}^l$ suit les étapes suivantes :

1. Calcul des $r(j)$: $\forall j\in\{1,\dots,n-1\}$, soit $r(j)=\#\{i\in\Pi_j\,/\,w_i=\perp\}$;
2. Vérification du taux de réponse : si $\sum r(j)>(1-\alpha)\,l$, le traçage est abandonné ;
3. Si $\forall j\in\{1,\dots,n-1\}$, $r(j)\leqslant d-\beta\,d$, le traçage s'appuie sur les valeurs :
   - Calcul de $c(j)$: pour tout $j\in\{1,\dots,n-1\}$, soit $S_j\subset\Pi_j$ tel que $\#S_j=\beta\,d$ et $\forall i\in S_j$, $w_i\neq\perp$ (on prend les premières positions de Π_j qui conviennent) ; on définit $c(j)$ par $\#\{i\in S_j\,/\,w_i=1\}$;
   - Calcul de l'ensemble : l'ensemble retourné par la procédure de traçage contient l'élément 1 si $c(1)>0$, l'élément n si $c(n-1)<\beta\,d$, et l'ensemble des $u\in\{2,\dots,n-1\}$ vérifiant $|c(u)-c(u-1)|>2\sqrt{(c(u)+c(u-1))\log(n/p)}$;

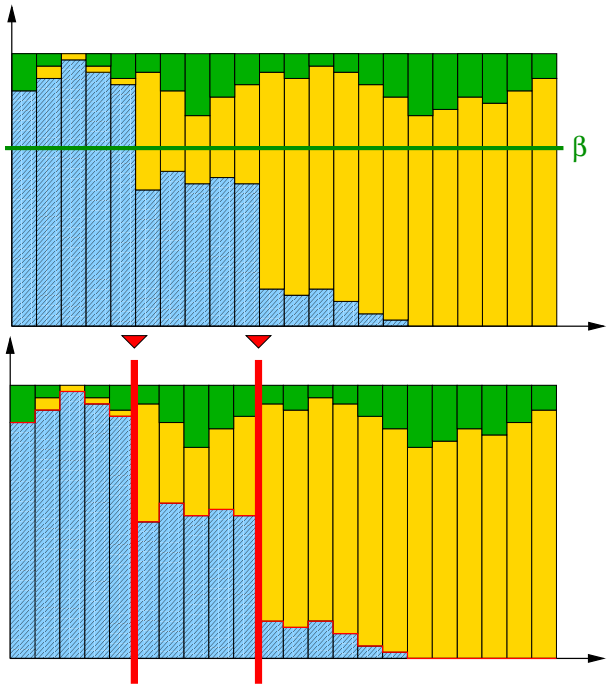

FIG. 4.7 – Effacement régulier : traçage basé sur les valeurs.

4. Sinon, le traçage s'appuie sur les effacements : l'ensemble retourné par la procédure de traçage contient l'ensemble des $u \in \{2, \ldots, n-1\}$ vérifiant $|r(u) - r(u-1)| > 2\sqrt{(r(u) + r(u-1)) \log(n/p)}$.

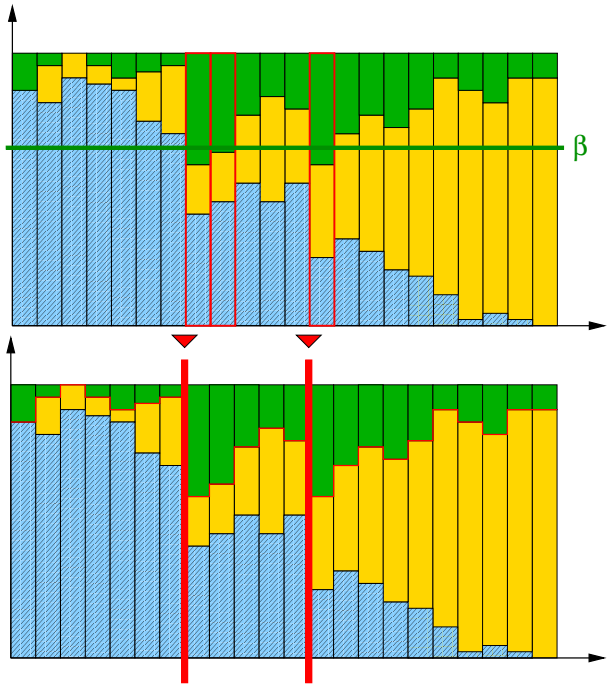

FIG. 4.8 – Effacement irrégulier : traçage basé sur les effacements.

Ainsi, le paramètre β sert à choisir le support du traçage (valeurs du mot dans les blocs, ou effacements du mot dans les blocs). Les figures 4.7 et 4.8 illustrent ces deux types de traçage utilisés.

4.4.5.4 Preuve de sécurité

Le théorème suivant donne des bornes minimales au paramètre d : lorsque le paramètre d est supérieur à ces bornes, la procédure de traçage est pertinente, pour toute coalition, et pour un taux de réponse supérieur ou égal à α.

Théorème 4.3 *Pour n et p vérifiant $n \geqslant 8p$, pour $\alpha \in]0,1]$, pour $\beta \in]0,\alpha]$ multiple entier de $1/d$, la famille de $(nd-d,n)$-codes binaires de type $\Gamma_0'(n,d,\alpha)$, avec la procédure de traçage $\tau'_{(p,\alpha,\beta)}$ décrite précédemment, est (p,n,α)-sûre contre des coalitions avec effacements lorsque*

$$\begin{cases} d > \frac{2\,(n-1)(n-2)\log(n/p)}{\beta} \sim \frac{2n^2\log(n/p)}{\beta}\,; \\ d \geqslant \frac{8\,(1-\beta)(n-1)(n-2)\log(n/p)}{(\alpha-\beta)^2} \sim \frac{8(1-\beta)\,n^2\log(n/p)}{(\alpha-\beta)^2}\,. \end{cases}$$

Remarque 4.3 *Pour la procédure de traçage, on peut utiliser n'importe quelle valeur de* $\beta \in]0, \alpha]$ *qui soit un multiple entier de* $1/d$*. L'intérêt est cependant d'optimiser simultanément les bornes inférieures requises sur* d *dans le théorème 4.3. Comme la première borne décroît lorsque* β *augmente, et que la second décroît lorsque* β *diminue, la valeur optimale pour* β *est intermédiaire. Plus précisément, on prend pour* β *le multiple entier de* $1/d$ *le plus proche de*

$$\frac{\alpha+2}{5} - \frac{2\sqrt{1+\alpha-\alpha^2}}{5}.$$

En pratique, la valeur de α dépend de choix techniques, et est par conséquent imposée. De cette valeur, on déduit par la remarque précédente la valeur de β, et une borne inférieure pour d. On ajuste ensuite les valeurs de β et de d pour obtenir simultanément la validité des inégalités et une valeur entière du produit $\beta\, d$.

Preuve La preuve se subdivise en deux parties, suivant le support du traçage utilisé. Dans chacune de ces parties, on procède exactement comme dans la preuve du théorème 4.2 : on prouve que la réponse de la procédure de traçage contient un élément hors de la coalition avec une probabilité inférieure à p, puis on prouve que cette réponse est nécessairement non-vide, sous contrainte d'une valeur de d suffisamment importante.

Traçage sur les valeurs. De la même manière que dans la preuve précédente, l'analyse probabiliste montre que pour un élément u hors de la coalition, les positions appartenant à S_{u-1} sont indistinguables de celles appartenant à S_u. Les résultats sont ainsi similaires à ceux de la preuve précédente, à la différence que le traçage s'appuie sur des blocs de βd positions, au lieu de d :

- la probabilité qu'un élément hors de la coalition soit identifié par la procédure de traçage est majorée par p,
- lorsque $\beta d > 2(n-1)(n-2)\log(n/p)$, la procédure de traçage ne répond jamais par un ensemble vide.

Traçage sur les effacements. Dans ce cas, on utilise également le même raisonnement que dans la preuve précédente, mais en considérant les effacements via les $r(j)$ plutôt que les valeurs 1 via les $c(j)$: pour un élément u hors de la coalition, les positions appartenant à $\Pi(u-1)$ sont indistinguables de celles appartenant à $\Pi(u)$, même en ce qui concerne les effacements. La probabilité qu'un élément hors de la coalition soit identifié par la procédure de traçage est donc majorée par p.

La vérification du caractère non-vide de l'ensemble renvoyé par la procédure de traçage nécessite par contre de petits ajustements. Dans un premier temps, on effectue la même transformation, en définissant les $h(j)$:

$$\forall j \in \{1, \ldots, n-1\},\ h(j) = 1/4 + r(j)/(2\log(n/p)).$$

On raisonne par l'absurde et on suppose que la procédure de traçage renvoie l'ensemble vide. On en déduit comme dans la preuve précédente que :

$$\forall j \in \{2, \ldots, n-1\},\ |\sqrt{h(j)} - \sqrt{h(j-1)}| \leqslant 1.$$

On en déduit que : $\forall (j_1, j_2) \in \{1, \ldots, n-1\}^2$, $|\sqrt{h(j_1)} - \sqrt{h(j_2)}| \leqslant n-2$. Or par hypothèse, il existe un j_1 pour lequel $r(j_1) \leqslant d - \alpha d$ (le mot w a un taux de réponse supérieur ou égal à α), et un j_2 pour lequel $r(j_2) > d - \beta d$ (sinon, on ne serait pas dans ce mode de traçage). D'où :

$$\sqrt{\frac{1}{4} + \frac{(1-\beta)\,d}{2\log(n/p)}} < \sqrt{\frac{1}{4} + \frac{(1-\alpha)\,d}{2\log(n/p)}} + n - 2.$$

On en tire les inégalités successives suivantes :

$$\frac{(\alpha-\beta)\,d}{2\log(n/p)} < (n-2)\left(\sqrt{\frac{1}{4} + \frac{(1-\beta)\,d}{2\log(n/p)}} + \sqrt{\frac{1}{4} + \frac{(1-\alpha)\,d}{2\log(n/p)}}\right).$$

$$\frac{(\alpha-\beta)\,d}{2\log(n/p)} < 2\,(n-2)\sqrt{\frac{1}{4} + \frac{(1-\beta)\,d}{2\log(n/p)}}.$$

$$\left(\frac{(\alpha-\beta)\,d}{2\log(n/p)}\right)^2 < (n-2)^2 + \frac{2d\,(1-\beta)(n-2)^2}{\log(n/p)}.$$

$$\left(\frac{(\alpha-\beta)\,d}{2\log(n/p)} - \frac{2\,(1-\beta)(n-2)^2}{(\alpha-\beta)}\right)^2 < (n-2)^2 + \left(\frac{2\,(1-\beta)(n-2)^2}{(\alpha-\beta)}\right)^2.$$

$$d < \frac{2\,(n-2)\log(n/p)}{(\alpha-\beta)}\left(\frac{4\,(1-\beta)(n-2)}{(\alpha-\beta)} + 1\right).$$

$$d < \frac{8\,(1-\beta)(n-1)(n-2)\log(n/p)}{(\alpha-\beta)^2}.$$

Lorsque d est supérieur ou égal à $8\,(1-\beta)(n-1)(n-2)\log(n/p)/(\alpha-\beta)^2$, l'hypothèse initiale d'une procédure de traçage retournant un ensemble vide est nécessairement infondée, ce qui conclut la preuve dans ce second cas. □

4.4.5.5 Performances

La famille de codes $\Gamma'_0(n, d, \alpha)$ a des performances très proches de celles de la famille de codes $\Gamma_0(n, d)$: la longueur des codes est cubique en n, et logarithmique en p. L'introduction d'effacements induit simplement un facteur multiplicatif, dépendant uniquement du taux de réponse minimum, α. Lorsque α tend vers 0, la longueur des codes augmente de manière quadratique en $1/\alpha$: pour des taux de réponse trop faibles, la famille de codes binaires $\Gamma'_0(n, d, \alpha)$ perd significativement en efficacité.

4.4.6 Code $\Gamma(n, d, \rho, t)$: coalitions limitées avec effacements

L'objectif est désormais de se limiter à des coalitions de taille limitée, et de rechercher une famille de codes offrant des performances tirant parti de ce cadre restreint. Pour cela, on utilise la stratégie donnée dans [BS95, FN93] pour construire une famille de codes dont la longueur est logarithmique en le nombre n de mots, à partir de la famille $\Gamma'_0(n, d, \alpha)$.

4.4.6.1 Définition

Il s'agit en fait de juxtaposer plusieurs codes sûrs contre des coalitions quelconques, mais réduits à $2t$ mots. Un mot du code global est donc la concaténation de mots pour chacun de ces codes, le choix du mot pour chacun de ces codes étant issu d'une répartition aléatoire équilibrée.

Un $((2t-1)d\rho, n)$-code de la famille $\Gamma(n, d, \rho, t)$ est donc généré de la manière suivante, à partir d'une famille $\Gamma'_0(n, d, \alpha)$:

- Pour tout $j \in \{1, \ldots, \rho\}$, on considère un code de la famille $\Gamma'_0(2t, d, \alpha)$, représenté par $\{w^{(j,1)}, \ldots, w^{(j,2t)}\} \subset \{0,1\}^{(2t-1)d}$. Ces codes sont générés en utilisant des partitions différentes de l'ensemble $\{1, \ldots, (2t-1)d\}$.
- Pour tout $j \in \{1, \ldots, \rho\}$, on considère une fonction h_j construite aléatoirement, de l'ensemble $\{1, \ldots, n\}$ dans $\{1, \ldots, 2t\}$: pour tout u appartenant à $\{1, \ldots, n\}$, $h_j(u)$ est choisi aléatoirement et uniformément dans l'ensemble $\{1, \ldots, 2t\}$.
- Pour tout u dans l'ensemble $\{1, \ldots, n\}$, le mot de code $w^{(u)}$ est la concaténation des mots élémentaires $(w^{(j,h_j(u))})_{1 \leqslant j \leqslant \rho}$.

4.4.6.2 Procédure de traçage

La procédure de traçage associée à un code de la famille $\Gamma(n, d, \rho, t)$ s'appuie sur les paramètres α et β (paramètres de traçage des codes de la famille $\Gamma'_0(2t, d, \alpha)$), γ (taux de réponse minimal) et p (probabilité d'erreur acceptée).

Elle consiste à découper le mot w en ρ morceaux, puis à tracer chacun de ces morceaux par le biais de la procédure de traçage associé au $j^{\text{ème}}$ code de la famille $\Gamma'_0(2t, d, \alpha)$, en utilisant le paramètre α, le paramètre β optimal, et la probabilité d'erreur $p/(2\rho)$. Pour chacun de ces traçages, on conserve une unique réponse k_j, tirée aléatoirement dans l'ensemble identifié (si l'ensemble est vide, on pose $k_j = 0$). Cette procédure répond alors par un singleton $\{u\}$, tel que $h_j(u) = k_j$ pour le plus grand nombre de j parmi les premières $\lceil (\gamma - \alpha)\rho/(1-\alpha) \rceil$ valeurs de j pour lesquelles $k_j \neq 0$ (en cas d'égalité, la procédure choisit sa réponse aléatoirement et uniformément parmi les ex-aequo).

4.4.6.3 Preuve de sécurité

Le théorème suivant donne une borne inférieure logarithmique en n à la longueur des codes de la famille $\Gamma(n, d, \rho, t)$ suffisante pour assurer la sécurité de cette famille face à des coalitions de taille inférieure ou égale à t.

Théorème 4.4 *Si la famille de codes $\Gamma_0(2t, d)$ est $(p/(2\rho), 2t, \alpha)$-sûre contre des coalitions avec effacements, alors pour tout $\gamma > \alpha$, la famille $\Gamma(n, d, \rho, t)$ de $((2t-1)d\rho, n)$-codes est (p, t, γ)-sûre contre des coalitions avec effacements à la condition que :*

$$\rho \geqslant \frac{4\,(1-\alpha)\,t\,\log(4n/p)}{(\gamma-\alpha)\log(3/2)}.$$

Remarque 4.4 *Pour une utilisation pratique, on a besoin de procéder en sens inversé par rapport au résultat donné dans le théorème 4.4 : γ est un paramètre du système, tout comme n et d, et on doit choisir la valeur de $\alpha < \gamma$ de manière optimale pour la longueur des codes de la famille.*

En négligeant l'influence des facteurs logarithmiques, ceci revient à chercher la valeur de α minimisant le quotient $(1-\alpha)/\big(\beta\,(\gamma-\alpha)\big)$, où β est défini par rapport à α selon la remarque 4.3. Lorsque γ tend vers 0, la valeur optimale de α est donc de l'ordre de $2\gamma/3$.

Preuve On considère un code de la famille $\Gamma(n, d, \rho, t)$, et un mot w issu d'une coalition C de taille inférieure ou égale à t sur ce code, avec un taux de réponse supérieur ou égal à γ. Soient $(w(j))_{1\leqslant j\leqslant\rho}$ les morceaux du mots $w \in \{0, 1, \perp\}^l$ obtenues après découpage. Pour tout $j \in \{1, \ldots, \rho\}$, on note $a(j)$ le taux de réponse sur le mot $w(j)$:

$$a(j) = \frac{\#\{i \in \{1, \ldots, (2t-1)d\} \,/\, w(j)_i \neq \perp\}}{(2t-1)d}.$$

La somme des $a(j)$ est par définition supérieure ou égale à $\gamma\,\rho$, et donc :

$$\#\{j \in \{1, \ldots, \rho\} \,/\, a(j) \geqslant \alpha\} + \alpha\,\big(\rho - \#\{j \in \{1, \ldots, \rho\} \,/\, a(j) \geqslant \alpha\}\big) \geqslant \gamma\,\rho.$$

On en déduit immédiatement : $\#\{j \in \{1, \ldots, \rho\} \,/\, a(j) \geqslant \alpha\} \geqslant \left\lceil \dfrac{(\gamma-\alpha)\,\rho}{1-\alpha} \right\rceil = \rho'$.

Il y a donc au moins ρ' morceaux $w(j)$ de w qui présentent un taux de réponse supérieur ou égal à α, et qui peuvent par conséquent être tracés en s'appuyant sur la famille $\Gamma_0(2t, d)$ de codes. La probabilité qu'il y ait une erreur durant l'une de ces procédures de traçage est majorée par $\rho\,(p/(2\rho)) \leqslant p/2$.

On suppose désormais que le traçage des morceaux $w(j)$ s'est déroulé sans erreur : il existe par conséquent un membre de la coalition $v \in C$ vérifiant $x_j = h_j(v)$ pour au moins $\lceil \rho'/t \rceil$ morceaux $w(j)$ parmi les premiers ρ' morceaux tels que $x_j \neq 0$.

On considère désormais un élément u hors de la coalition C. Pour tout $j \in \{1, \ldots, \rho\}$ tel que $x_j \neq 0$, l'égalité $h_j(u) = x_j$ a lieu avec probabilité $1/(2t)$, puisque $h_j(u)$ est choisi aléatoirement dans $\{1, \ldots, 2t\}$. On note $d(u)$ le nombre de morceaux vérifiant $h_j(u) = x_j$ parmi les ρ' premiers morceaux tels que $x_j \neq 0$. Cette variable aléatoire vérifie l'équation suivante :

$$\forall m \in \{0, \ldots, \rho' - 1\}, \quad \Pr[d(u) = m+1] = \frac{\rho' - m}{(2t-1)(m+1)} \Pr[d(u) = m].$$

On suppose que la borne du théorème est respectée : $\rho' \geqslant 4\,t\log(4n/p)/\log(3/2)$. En particulier, on a l'inégalité $\rho' \geqslant 4\,t$. Lorsque $m \geqslant \lfloor 3\rho'/(4t)\rfloor$, on obtient la borne :

$$\Pr[d(u) = m+1] \leqslant \frac{(4t-3)\rho' + 4t}{3(2t-1)\rho'} \Pr[d(u) = m] \leqslant \frac{2}{3} \Pr[d(u) = m].$$

Ainsi, $\Pr\left[d(u) = \lceil \rho'/t \rceil\right] \leqslant (2/3)^{\rho'/(4t)} \Pr\left[d(u) = \lfloor 3\rho'/(4t)\rfloor\right] \leqslant p/(4n)$.

De plus, lorsque $m \geqslant \rho'/t$,

$$\Pr[d(u) = m+1] \leqslant \frac{t-1}{2t-1} \Pr[d(u) = m] \leqslant \frac{1}{2} \Pr[d(u) = m].$$

On en déduit : $\Pr\left[d(u) \geqslant \lceil \rho'/t \rceil\right] \leqslant 2 \Pr\left[d(u) = \lceil \rho'/t \rceil\right] \leqslant p/(2n)$.

Lorsque les morceaux $w(j)$ sont tracés sans erreur, la probabilité que la procédure de traçage globale retourne un utilisateur hors de la coalition est donc bornée par $p/2$. Globalement, la procédure de traçage échoue donc avec une probabilité inférieure ou égale à p. □

4.4.6.4 Performances

La famille de codes $\Gamma(n, d, \rho, t)$ est similaire à la famille de codes présentée dans [BS95], à la différence près qu'elle tolère des effacements. Globalement, la longueur des codes de cette famille est :

$$\frac{16\,(1-\alpha)\,t\,(t-1)(2t-1)^2}{(\gamma-\alpha)\,\beta\log(3/2)} \log\left(\frac{4n}{p}\right) \log\left(\frac{16\,(1-\alpha)\,t^2\,\log(4n/p)}{(\gamma-\alpha)\,p\log(3/2)}\right).$$

La longueur des codes est logarithmique en n et en p, ce qui est le résultat recherché : une dépendance polynomiale en n est dissuasive.

En contrepartie, la longueur des codes a une dépendance de l'ordre de t^4 en le nombre de traîtres considérés, ce qui représente des longueurs considérables pour peu que l'on cherche à se protéger contre des coalitions non-restreintes à quelques utilisateurs.

4.5 Schéma complet

On dispose désormais des briques fondamentales permettant de reconstruire le schéma complet de traçage de traîtres : un schéma élémentaire de traçage de traîtres, sûr contre des décodeurs pirates forts, et un code sûr contre des coalitions avec effacements. Il suffit donc de combiner ces ingrédients à la manière de [KY02] pour obtenir le schéma recherché.

4.5.1 Définition

Définition 4.13 (Schéma complet de traçage de traîtres) *Soit* $\{w^{(1)}, \ldots, w^{(n)}\}$ *un* (l, n)*-code issu d'une famille de codes binaires sûre contre des coalitions avec effacements. Soit* (I, E, D, T) *un schéma élémentaire de traçage de traîtres pour deux utilisateurs. Le schéma complet de traçage de traîtres construit à partir du code et du schéma élémentaire est défini par ces algorithmes.*

- ***Initialisation.*** *Elle consiste en* l *appels à* I *: pour tout* $i \in \{1, \ldots, l\}$*, le* $i^{ème}$ *appel à* I *donne la clé de chiffrement* $ek^{(i)}$*, les clés de déchiffrement* $dk_1^{(i)}$ *et* $dk_2^{(i)}$ *et la clé de traçage* $tk^{(i)}$*. La clé globale de chiffrement est :* $\mathsf{ek} = (ek^{(1)}, \ldots, ek^{(l)})$*. La clé globale de traçage est :* $\mathsf{tk} = (tk^{(1)}, \ldots, tk^{(l)})$*. La clé de déchiffrement associée à l'utilisateur* u *est :* $\mathsf{dk}_u = (\mathsf{dk}(u, 1), \ldots, \mathsf{dk}(u, l))$*, où pour tout* i*,* $\mathsf{dk}(u, i)$ *est la clé* $dk_1^{(i)}$ *si* $w_i^{(u)} = 0$*,* $dk_2^{(i)}$ *si* $w_i^{(u)} = 1$*.*
- ***Chiffrement.*** *Le chiffrement d'un message* $(m_1, \ldots, m_l)$ *est la concaténation des chiffrés des* m_i *par chacune des instances du schéma élémentaire :*

$$(E_{ek^{(1)}}(m_1), \ldots, E_{ek^{(l)}}(m_l)).$$

- ***Déchiffrement.*** *Le déchiffrement d'un chiffré* $(c_1, \ldots, c_l)$ *est la concaténation des déchiffrés des* c_i *par chacune des instances du schéma élémentaire :*

$$(D_{\mathsf{dk}(u,1)}(c_1), \ldots, D_{\mathsf{dk}(u,l)}(c_l)).$$

- ***Traçage.*** *Le traçage consiste dans un premier temps à tracer toutes les instances du schéma élémentaire, en utilisant pour chaque instance la procédure* T *avec la clé* $tk^{(i)}$*. Si la clé de déchiffrement* $tk_1^{(i)}$ *est détectée, on pose* $w'_i = 0$*; si au contraire c'est la clé* $tk_2^{(i)}$ *qui est détectée, on pose* $w'_i = 1$*. Si le traçage n'est pas possible, on insère un effacement. Le mot* w' *est ensuite tracé dans le cadre de la procédure de traçage de la famille de codes sûrs contre des coalitions. Le résultat de cet ultime traçage donne l'utilisateur à incriminer.*

4.5.2 Preuve de sécurité

La sécurité de cette construction est donnée dans le théorème suivant :

Théorème 4.5 *Si le schéma élémentaire de traçage de traîtres est* $(p_b, 2, \alpha_b)$*-sûr, si le* (n, l)*-code utilisé est issu d'une famille de codes binaires* (p_c, t, α_c)*-sûre contre des coalitions avec effacements, alors la construction présentée en définition 4.5 est un schéma de traçage de traîtres* (p, t, α)*-sûr pour* n *utilisateurs, où :*

$$\alpha = 1 - (1 - \alpha_b)(1 - \alpha_c) \quad \text{et} \quad p = p_c + l\, p_b.$$

Preuve Soit $\mathcal{P}$ un décodeur pirate fort qui déchiffre des messages chiffrés valides avec une probabilité supérieure ou égale à α. Pour toute instance $i \in \{1, \ldots, l\}$ du schéma

élémentaire de traçage de traîtres, on définit $\alpha(i)$ le taux de réponse de $\mathcal{P}$ sur l'instance d'indice i. On a nécessairement : $\sum_{i=1}^{l} \alpha(i) \geqslant \alpha\, l$, d'où :

$$\#\{i \in \{1,\ldots,l\} \,/\, \alpha(i) \geqslant \alpha_b\} + \alpha_b\,(l - \#\{i \in \{1,\ldots,l\} \,/\, \alpha(i) \geqslant \alpha_b\}) > \alpha\, l.$$

On en déduit : $\#\{i \in \{1,\ldots,l\} \,/\, \alpha(i) \geqslant \alpha_b\} > \dfrac{\alpha - \alpha_b}{1 - \alpha_b}\, l = \alpha_c\, l.$

Lors du jeu de traçage, le décodeur pirate reçoit les clés de déchiffrement en fonction des mots $w^{(u)}$ qui correspondent aux membres de la coalition. La procédure de traçage permet de reconstruire un mot $w' \in \{0,1,\perp\}^l$ à partir des réponses fournies par $\mathcal{P}$. Le mot w' présente un taux minimum de réponse égal à α_c, ce qui est conforme aux conditions d'utilisation de la procédure de traçage de la famille de codes sûre contre des coalitions avec effacement : cette procédure permet d'identifier un utilisateur. On note $F_{\alpha_c}(C)$ l'ensemble des mots qu'un adversaire du code avec un taux de réponse minimum α_c peut construire à partir de la coalition C.

Premier cas : $w' \notin F_{\alpha_c}(C)$. Dans ce cas, on utilise le décodeur pirate $\mathcal{P}$ pour construire un décodeur pirate $\mathcal{P}_b$ du schéma élémentaire de traçage de traîtres : $\mathcal{P}_b$ génère $l-1$ instances du schéma élémentaire du traçage de traîtres, et ajoute l'instance sur laquelle il est défié à ces $l-1$ instances. Le décodeur pirate $\mathcal{P}$ joue donc sur $l-1$ instances arbitraires, que $\mathcal{P}_b$ contrôle complètement, et sur une dernière instance qui constitue le défi de $\mathcal{P}_b$. Le mot w' n'appartient pas à l'ensemble $F_{\alpha_c}(C)$. Il y a donc au moins une position pour laquelle le traçage de l'instance a échoué. Avec au moins une chance sur l, il s'agit de l'instance sur laquelle $\mathcal{P}_b$ a été défié. Comme le schéma élémentaire de traçage de traîtres est supposé $(p_b, 2, \alpha_b)$-sûr, la probabilité que w' n'appartienne pas à $F_{\alpha_c}(C)$ est donc bornée par $l\, p_b$.

Second cas : $w' \in F_{\alpha_c}(C)$. Dans ce cas, on utilise le décodeur pirate $\mathcal{P}$ pour construire un adversaire $\mathcal{A}_c$ de la famille de codes (p_c, t, α_c)-sûre contre les coalitions avec effacements. L'adversaire $\mathcal{A}_c$ reçoit des mots du code, et se contente de transposer cette information sous la forme de clés de déchiffrement qui sont données à $\mathcal{P}$. Lorsque la procédure de traçage échoue sur $\mathcal{P}$, $\mathcal{A}_c$ identifie le mot w' utilisé implicitement par $\mathcal{P}$, et répond avec ce mot w' : par hypothèse, w' est dans l'ensemble des mots que $\mathcal{A}_c$ peut utiliser, et si $\mathcal{P}$ gagne le jeu de traçage, alors nécessairement $\mathcal{A}_c$ gagne le jeu de marquage. Dans ce second cas, on en déduit que la probabilité que $\mathcal{P}$ gagne le jeu de traçage alors que le mot obtenu au cours du traçage appartient à $F_{\alpha_c}(C)$ est inférieur ou égal à p_c.

Globalement, on obtient donc : $\Pr[\mathcal{P} \text{ gagne le jeu de traçage}] \leqslant p_c + l\, p_b$. On en déduit que le schéma global de traçage de traîtres est effectivement (p, t, α)-sûr. □

4.5.3 Performances

Tout l'intérêt de cette construction réside dans le fait qu'elle a le même taux de chiffrement que le schéma élémentaire de traçage de traîtres. En l'occurrence, le schéma

élémentaire proposé en partie 4.3 a un taux de chiffrement constant, qui peut être pris égal à 2 (notamment en utilisant le chiffrement hybride).

En revanche, toutes les clés (chiffrement, déchiffrement, traçage) ont une longueur multipliée par l par rapport au schéma élémentaire. C'est le cas pour la quantité de données à transmettre durant la procédure traçage.

La taille des messages chiffrés et des messages clairs est également multipliée par l, mais là, il est possible de chiffrer ou de déchiffrer les données progressivement, en traitant les instances du schéma élémentaire indépendamment les unes des autres : c'est un atout considérable, dans la mesure où le facteur multiplicatif est important, et où cette fonctionnalité était auparavant inexistante.

En termes de confidentialité du contenu, le schéma complet de traçage de traîtres réalise le même niveau de sécurité que le schéma élémentaire, dans le cadre d'attaques à messages choisis (LoR-CPA par exemple).

4.6 Conclusion

Dans cette partie, on a abordé le problème du traçage de traîtres par la recherche d'un modèle de sécurité prenant en compte divers moyens relativement simples pour un décodeur pirate d'éviter le traçage. Ainsi, un décodeur pirate dispose d'une mémoire, et peut ne pas répondre à certaines requêtes ou se bloquer lorsqu'un traçage est détecté. Ces quelques techniques, relativement faciles à mettre en place dans la conception d'un décodeur pirate, sont relativement difficiles à contourner pour permettre un traçage efficace des concepteurs.

Une fois le modèle élaboré, on a constaté que les schémas permettant un traçage dans ces circonstances n'avaient pas une efficacité suffisante pour permettre une utilisation pratique. On a alors construit un nouveau schéma, sûr dans ce modèle, respectant un ratio constant entre les tailles d'un chiffré et d'un clair associé.

Le modèle de sécurité impose l'utilisation du marquage de données, qui force au moins une duplication d'un message clair émis. Au niveau du ratio précédemment évoqué, on ne peut donc espérer obtenir mieux qu'un facteur 2, et l'utilisation d'un couplage semble donc inutile.

La taille des clés de déchiffrement peut également être considérée comme un critère pertinent de l'efficacité du schéma. Le schéma présenté ici nécessite des clés de déchiffrement très longues, comme dans les schémas précédents de traçage de traîtres avec un ratio constant entre les tailles d'un chiffré et d'un clair associé. Cette taille des clés de déchiffrement peut être améliorée, sans doute significativement, par la recherche de meilleurs codes sûrs contre des coalitions avec effacement : le code présenté montre seulement l'existence de tels codes, avec une longueur ne dépendant que logarithmiquement du nombre d'utilisateurs, mais il est sans doute possible de construire des codes plus performants. Une fois un code plus efficace construit, le théorème 4.5 donne directement un schéma de traçage de traîtres amélioré.

Cette taille relativement importante des clés de déchiffrement implique une taille également importante des messages clairs et chiffrés à transmettre. Cependant, l'utilisation de codes particuliers, plus robustes que ceux utilisés jusqu'à présent, permet d'accepter un déchiffrement progressif, c'est-à-dire que le message clair associé à un message chiffré peut être déterminé au fur et à mesure de la réception du message chiffré, sans nécessiter la connaissance préalable de l'ensemble du message chiffré. Cette fonctionnalité, nouvelle dans le cadre de schémas de traçage de traîtres à ratio constant, est néanmoins très importante en pratique, lorsque la taille des chiffrés nécessite un temps de transmission complète important.

Bibliographie

[ACD+06] Michel Abdalla, Dario Catalano, Alexander W. Dent, John Malone-Lee, Gregory Neven et Nigel P. Smart : Identity-based encryption gone wild. *In Proc. of ICALP'06*, pages 300–311, 2006.

[AD93a] Leonard M. Adleman et Jonathan Demarrais : A subexponential algorithm for discrete logarithms over all finite fields. *In Proc. of Advances in Cryptology – Crypto'93*, pages 147–158, 1993.

[AD93b] Leonard M. Adleman et Jonathan Demarrais : A subexponential algorithm for discrete logarithms over all finite fields. *Mathematics of Computation*, 61(203):1–15, 1993.

[Adl94] Leonard M. Adleman : The function field sieve. *In Proc. of the 1st Algorithmic Number Theory Symposium (ANTS-I)*, volume 877, pages 108–121, 1994.

[AF07] Masayuki Abe et Serge Fehr : Perfect NIZK with adaptive soundness. *In Proc. of Advances in Cryptology – TCC'07*, pages 118–136, 2007.

[BB04a] Dan Boneh et Xavier Boyen : Efficient selective-ID secure identity-based encryption without random oracles. *In Proc. of Advances in Cryptology – Eurocrypt'04*, pages 223–238, 2004.

[BB04b] Dan Boneh et Xavier Boyen : Short signatures without random oracles. *In Proc. of Advances in Cryptology – Eurocrypt'04*, pages 56–73, 2004.

[BBG05] Dan Boneh, Xavier Boyen et Eu-Jin Goh : Hierarchical identity based encryption with constant size ciphertext. *In Proc. of Advances in Cryptology – Eurocrypt'05*, pages 440–456, 2005.

[BDJR97] Mihir Bellare, Anand Desai, Eron Jokipii et Phillip Rogaway : A concrete security treatment of symmetric encryption. *In Proc. of Foundations of Computer Science (FOCS'97)*, pages 394–403, 1997.

[BDNS06] James Birkett, Alexander W. Dent, Gregory Neven et Jacob Schuldt : Efficient chosen-ciphertext secure identity-based encryption with wildcards. Rapport technique 2006/377, Cryptology ePrint Archive, 2006.

[Ber68] Elwyn R. Berlekamp : *Algebraic Coding Theory.* McGraw-Hill, New York, 1968.

[BF99] Dan Boneh et Matthew Franklin : An efficient public key traitor tracing scheme. *In Proc. of Advances in Cryptology – Crypto'99*, pages 338–353, 1999.

[BF01] Dan Boneh et Matthew Franklin : Identity-based encryption from the Weil pairing. *In Proc. of Advances in Cryptology – Crypto'01*, pages 213–229, 2001.

[BGW05] Dan Boneh, Craig Gentry et Brent Waters : Collusion resistant broadcast encryption with short ciphertexts and private keys. *In Proc. of Advances in Cryptology – Crypto'05*, pages 258–275, 2005.

[BGY80] Richard P. Brent, Fred G. Gustavson et David Y. Y. Yun : Fast solution of Toeplitz systems of equations and computation of Padé approximants. *Journal of Algorithms*, 1:259–295, 1980.

[BMSS07] Alin Bostan, François Morain, Bruno Salvy et Éric Schost : Fast algorithms for computing isogenies between elliptic curves. *Mathematics of Computation*, 2007. À paraître.

[BR93] Mihir Bellare et Phillip Rogaway : Random oracles are practical : A paradigm for designing efficient protocols. *In Proc. of ACM-CCS'93*, pages 62–73, 1993.

[Bro05] Daniel R. L. Brown : Generic groups, collision resistance, and ECDSA. *Designs, Codes and Cryptography*, 35(1):119–152, 2005.

[BS95] Dan Boneh et James Shaw : Collusion-secure fingerprinting for digital data. *In Proc. of Advances in Cryptology – Crypto'95*, pages 452–465, 1995.

[BSW06] Dan Boneh, Amit Sahai et Brent Waters : Fully collusion resistant traitor tracing with short ciphertexts and private keys. *In Proc. of Advances in Cryptology – Eurocrypt'06*, pages 573–592, 2006.

[BSW07] John Bethencourt, Amit Sahai et Brent Waters : Ciphertext-policy attribute-based encryption. *In Proc. of IEEE Symposium on Security and Privacy*, pages 321–334, 2007.

[BW06] Dan Boneh et Brent Waters : A fully collusion resistant broadcast, trace, and revoke system. *In Proc. of ACM-CCS'06*, pages 211–220, 2006.

[CFA+05] Henri Cohen, Gerhard Frey, Roberto M. Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen et Fré Vercauteren : *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. Discrete Mathematics and Its Applications. Chapman and Hall / CRC, 2005.

[CFN94] Benny Chor, Amos Fiat et Moni Naor : Tracing traitors. *In Proc. of Advances in Cryptology – Crypto'94*, pages 257–270, 1994.

[CFNP00] Benny Chor, Amos Fiat, Moni Naor et Benny Pinkas : Tracing traitors. *IEEE Transactions on Information Theory*, 46(3):893–910, 2000.

[CGI+99] Ran Canetti, Juan Garay, Gene Itkis, Daniele Micciancio, Moni Naor et Benny Pinkas : Multicast security : A taxonomy and efficient constructions. *IEEE Infocom'99*, 2:708–716, 1999.

[CKLS96] Ingemar J. Cox, Joe Killian, Tom Leighton et Talal Shamoon : A secure, robust watermark for multimedia. *In Proc. of Information Hiding (IH'96)*, pages 257–270, 1996.

[CMN99] Ran Canetti, Tal Malkin et Kobbi Nissim : Efficient communication-storage tradeoffs for multicast encryption. *In Proc. of Advances in Cryptology – Eurocrypt'99*, pages 459–474, 1999.

[Cou94] Jean-Marc Couveignes : *Quelques calculs en théorie des nombres*. Thèse de doctorat, Université de Bordeaux I, 1994.

[Cou96] Jean-Marc Couveignes : Computing l-isogenies with the p-torsion. *In Proc. of the 2nd Algorithmic Number Theory Symposium (ANTS-II)*, volume 1122, pages 59–65, 1996.

[Cou00] Jean-Marc Couveignes : Isomorphisms between Artin-Schreier towers. *Mathematics of Computation*, 69(232):1625–1631, 2000.

[CPP05] Herve Chabanne, Duong Hieu Phan et David Pointcheval : Public traceability in traitor tracing schemes. *In Proc. of Advances in Cryptology – Eurocrypt'05*, pages 542–558, 2005.

[Den02] Alexander W. Dent : Adapting the weaknesses of the random oracle model to the generic group model. *In Proc. of Advances in Cryptology – Asiacrypt'02*, pages 100–109, 2002.

[DF02] Yevgeniy Dodis et Nelly Fazio : Public key broadcast encryption for stateless receivers. *In Proc. of Security and Privacy in Digital Right Management (DRM'02)*, pages 61–80, 2002.

[DH76] Whitfield Diffie et Martin E. Hellman : New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[Dor87] Jean-Louis Dornstetter : On the equivalence between Berlekamp's and Euclid's algorithms. *IEEE Transactions on Information Theory*, 33(3):428–431, 1987.

[DPP07] Cecile Delerablee, Pascal Paillier et David Pointcheval : Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. *In Proc. of the First Conference on Pairing-based Cryptography (Pairing'07)*, 2007.

[EKK99] Funda Ergun, Joe Kilian et Ravi Kumar : A note on the limits of collusion-resistant watermarks. *In Proc. of Advances in Cryptology – Eurocrypt'99*, pages 140–149, 1999.

[Eng07] Andreas Enge : Computing modular polynomials in quasi-linear time. Rapport technique 0704.3177, e-print ArXiv.org Mathematics, 2007.

[FN93] Amos Fiat et Moni Naor : Broadcast encryption. *In Proc. of Advances in Cryptology – Crypto'93*, pages 480–491, 1993.

[FO99a] Eiichiro Fujisaki et Tatsuaki Okamoto : How to enhance the security of public-key encryption at minimum cost. *In Proc. of Advances in Cryptology – PKC'99*, pages 53–68, 1999.

[FO99b] Eiichiro Fujisaki et Tatsuaki Okamoto : Secure integration of asymmetric and symmetric encryption schemes. *In Proc. of Advances in Cryptology – Crypto'99*, pages 537–554, 1999.

[FT99] Amos Fiat et Tamir Tassa : Dynamic traitor tracing. *In Proc. of Advances in Cryptology – Crypto'99*, pages 354–371, 1999.

[Gam84] Taher El Gamal : A public key cryptosystem and a signature scheme based on discrete logarithms. *In Proc. of Advances in Cryptology – Crypto'84*, pages 10–18, 1984.

[Gam85] Taher El Gamal : A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[GHM03] Pierrick Gaudry, Guillaume Hanrot et Francois Morain : Cours de théorie algorithmique des nombres, DEA Algorithmique, Paris, 2003.

[Gor93] Daniel M. Gordon : Discrete logarithms in GF(p) using the number field sieve. *SIAM Journal on Discrete Mathematics (SIDMA)*, 6(1):124–138, 1993.

[GP99] Hans-Jurgen Guth et Birgit Pfitzmann : Error- and collusion-secure fingerprinting for digital data. *In Proc. of Information Hiding (IH'99)*, pages 134–145, 1999.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai et Brent Waters : Attribute-based encryption for fine-grained access control of encrypted data. *In Proc. of ACM-CCS'06*, pages 89–98, 2006.

[GST04] Michael T. Goodrich, Jonathan Z. Sun et Roberto Tamassia : Efficient tree-based revocation in groups of low-state devices. *In Proc. of Advances in Cryptology – Crypto'04*, pages 511–527, 2004.

[GSW00] Juan A. Garay, Jessica Staddon et Avishai Wool : Long-lived broadcast encryption. *In Proc. of Advances in Cryptology – Crypto'00*, pages 333–352, 2000.

[HS02] Dani Halevy et Adi Shamir : The LSD broadcast encryption scheme. *In Proc. of Advances in Cryptology – Crypto'02*, pages 47–60, 2002.

[JL06a] Antoine Joux et Reynald Lercier : Counting points on elliptic curves in medium characteristic. Rapport technique 2006/176, Cryptology ePrint Archive, 2006.

[JL06b] Antoine Joux et Reynald Lercier : The function field sieve in the medium prime case. *In Proc. of Advances in Cryptology – Eurocrypt'06*, pages 257–270, 2006.

[JLSV06] Antoine Joux, Reynald Lercier, Nigel Smart et Frederik Vercauteren : The function field sieve in the medium prime case. *In Proc. of Advances in Cryptology – Crypto'06*, pages 326–344, 2006.

[Jou00] Antoine Joux : A one round protocol for tripartite Diffie-Hellman. *In Proc. of the 4th Algorithmic Number Theory Symposium (ANTS-IV)*, volume 1838, pages 385–393, 2000.

[KD98] Kaoru Kurosawa et Yvo Desmedt : Optimum traitor tracing and asymmetric schemes. *In Proc. of Advances in Cryptology – Eurocrypt'98*, pages 145–157, 1998.

[KM07] Neal Koblitz et Alfred Menezes : Another look at generic groups. *Advances in Mathematics of Communications*, 1:13–28, 2007.

[KRS99] Ravi Kumar, Sridhar Rajagopalan et Amit Sahai : Coding constructions for blacklisting problems without computational assumptions. *In Proc. of Advances in Cryptology – Crypto'99*, pages 609–623, 1999.

[KY01a] Aggelos Kiayias et Moti Yung : On crafty pirates and foxy tracers. *In Proc. of Security and Privacy in Digital Right Management (DRM'01)*, pages 22–39, 2001.

[KY01b] Aggelos Kiayias et Moti Yung : Self protecting pirates and black-box traitor tracing. *In Proc. of Advances in Cryptology – Crypto'01*, pages 63–79, 2001.

[KY02] Aggelos Kiayias et Moti Yung : Traitor tracing with constant transmission rate. *In Proc. of Advances in Cryptology – Eurocrypt'02*, pages 450–465, 2002.

[Len04] Arjen K. Lenstra : Key lengths. *In Handbook of Information Security.* 2004.

[Ler96] Reynald Lercier : Computing isogenies in GF(2^n). *In Proc. of the 2nd Algorithmic Number Theory Symposium (ANTS-II)*, volume 1122, pages 197–212, 1996.

[LN83] Rudolf Lidl et Harald Niederreiter : *Finite Fields*, volume 20 de *Encyclopedia of Mathematics and its Applications.* Addison-Wesley, 1983.

[LR88] Michael Luby et Charles Rackoff : How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing (SICOMP)*, 17(2):373–386, 1988.

[LS08a] Reynald Lercier et Thomas Sirvent : On Elkies subgroups of ℓ-torsion points in elliptic curves defined over a finite field. *Journal de Théorie des Nombres de Bordeaux*, 2008. Soumis à Publication.

[LS08b] David Lubicz et Thomas Sirvent : Attribute-based broadcast encryption scheme made efficient. *In Proc. of Advances in Cryptology – Africacrypt'08*, pages 325–342, 2008.

[LS08c] David Lubicz et Thomas Sirvent : On generic groups and related bilinear problems. *In* M. Joye et G. Neven, éditeur : *Identity-Based Cryptography*, Cryptology and Information Security Series. IOS Press, 2008. À Paraître.

[LV00] Arjen K. Lenstra et Eric R. Verheul : Selecting cryptographic key sizes. *In Proc. of Advances in Cryptology – PKC'00*, pages 446–465, 2000.

[LV01] Arjen K. Lenstra et Eric R. Verheul : Selecting cryptographic key sizes. *Journal of Cryptology*, 14(4):255–293, 2001.

[Mas69] James L. Massey : Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, 15(1):122–127, 1969.

[MI04] Tatsuyuki Matsushita et Hideki Imai : A public-key black-box traitor tracing scheme with sublinear ciphertext size against self-defensive pirates. *In Proc. of Advances in Cryptology – Asiacrypt'04*, pages 260–275, 2004.

[Mor95] François Morain : Calcul du nombre de points sur une courbe elliptique dans un corps fini : aspects algorithmiques. *Journal de théorie des nombres de Bordeaux*, 7(1):255–282, 1995.

[MSK99] Shigeo Mitsunari, Ryuichi Sakai et Masao Kasahara : A new traitor tracing. *IEICE Trans.*, E82(1), Janvier 1999.

[MTI86] Tsutomu Matsumoto, Youichi Takashima et Hideki Imai : On seeking smart public-key-distribution systems. *IEICE Trans.*, E69-E(2), Février 1986.

[MvOV01] Alfred Menezes, Paul van Oorschot et Scott Vanstone : *Handbook of Applied Cryptography*. Discrete Mathematics and its Applications. CRC Press, 2001. Édition corrigée.

[N06] FIPS 186-3 : Digital signature standard (DSS). Rapport technique, U.S. Dept. of Commerce / National Technical Information Service (N.I.S.T.), 2006.

[Nec93] V. I. Nechaev : Complexity of a determinate algorithm for the discrete logarithm. *Mathematicheskie Zametki*, 55(2):91–101, 1993.

[NNL01] Dalit Naor, Moni Naor et Lotspiech : Revocation and tracing schemes for stateless receivers. *In Proc. of Advances in Cryptology – Crypto'01*, pages 41–62, 2001.

[NP98] Moni Naor et Benny Pinkas : Threshold traitor tracing. *In Proc. of Advances in Cryptology – Crypto'98*, pages 502–517, 1998.

[NSS04] David Naccache, Nigel Smart et Jacques Stern : Projective coordinates leak. *In Proc. of Advances in Cryptology – Eurocrypt'04*, pages 257–267, 2004.

[OP01] Tatsuaki Okamoto et David Pointcheval : React : Rapid enhanced-security asymmetric cryptosystem transform. *In Proc. of Cryptographers' Track, RSA Conference – CT-RSA'01*, pages 159–175, 2001.

[OSW07] Rafail Ostrovsky, Amit Sahai et Brent Waters : Attribute-based encryption with non-monotonic access structures. *In Proc. of ACM-CCS'07*, pages 195–203, 2007.

[Pan00] Victor Y. Pan : New techniques for the computation of linear recurrence coefficients. *Finite Fields and Their Applications*, 6(1):93–118, 2000.

[Pap94] Christos H. Papadimitriou : *Computational complexity*. Addison-Wesley, Reading, MA, 1994.

[PH78] Stephen C. Pohlig et Martin E. Hellman : An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Transactions on Information Theory*, 24(1):106–110, 1978.

[Pol78] J. M. Pollard : Monte Carlo methods for index computation (mod p). *Mathematics of Computation*, 32(143):918–924, 1978.

[PST01] Adrian Perrig, Dawn Song et J. D. Tygar : Elk, a new protocol for efficient large-group key distribution. *In Proc. of IEEE Symposium on Security and Privacy*, pages 247–262, 2001.

[Riv97] Ronald Rivest : All-or-nothing encryption and the package transform. *In Proc. of Fast Software Encryption'97*, pages 210–218, 1997.

[Sch80] J. T. Schwartz : Fast probabilistic algorithms for verification of polynomial identities. *J. Assoc. Comput. Mach.*, 27(4):701–717, 1980.

[Sch91] Claus P. Schnorr : Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[Sch95] René Schoof : Counting points on elliptic curves over finite fields. *Journal de théorie des nombres de Bordeaux*, 7(1):219–254, 1995.

[Sha79] Adi Shamir : How to share a secret. *Communications of the ACM archive*, 22(11):612–613, 1979.

[Sho89] Victor Shoup : *Removing Randomness from Computational Number Theory*. Thèse de doctorat, University of Winsconsin, 1989.

[Sho97] Victor Shoup : Lower bounds for discrete logarithms and related problems. *In Proc. of Advances in Cryptology – Eurocrypt'97*, pages 256–266, 1997.

[Sil86] J. H. Silverman : *The arithmetic of elliptic curves*, volume 106 de *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1986. Édition corrigée.

[Sir07] Thomas Sirvent : Traitor tracing scheme with constant ciphertext rate against powerful pirates. *In Proc. of Workshop on Coding and Cryptography (WCC'07)*, pages 379–388, 2007.

[SNW01] Reihaneh Safavi-Naini et Yejing Wang : Collusion secure q-ary fingerprinting for perceptual content. *In Proc. of Security and Privacy in Digital Right Management (DRM'01)*, pages 57–75, 2001.

[SNW02] Reihaneh Safavi-Naini et Yejing Wang : Traitor tracing for shortened and corrupted fingerprints. *In Proc. of Security and Privacy in Digital Right Management (DRM'02)*, pages 81–100, 2002.

[SPMLS02] Jacques Stern, David Pointcheval, John Malone-Lee et Nigel P. Smart : Flaws in applying proof methodologies to signature schemes. *In Proc. of Advances in Cryptology – Crypto'02*, pages 93–110, 2002.

[SW05] Amit Sahai et Brent Waters : Fuzzy identity-based encryption. *In Proc. of Advances in Cryptology – Eurocrypt'05*, pages 457–473, 2005.

[TSNZ03] V. D. To, R. Safavi-Naini et F. Zhang : New traitor tracing schemes using bilinear map. *In Proc. of Security and Privacy in Digital Right Management (DRM'03)*, pages 67–76, 2003.

[Vél71] Jacques Vélu : Isogénies entre courbes elliptiques. *Comptes Rendus de l'Académie des Sciences de Paris*, 273:238–241, 1971. Série A.

[WGL98] Chung Kei Wong, Mohamed Gouda et Simon S. Lam : Secure group communications using key graphs. *In Proc. of ACM-SIGCOMM'98*, pages 68–79, 1998.

[WHA99] Debby. M. Wallner, Eric J. Harder et Ryan C. Agee : Key management for multicast : Issues and architectures, RFC 2627, 1999.

[YW05] Tsz Hon Yuen et Victor K. Wei : Fast and proven secure blind identity-based signcryption from pairings. *In Proc. of Cryptographers' Track, RSA Conference – CT-RSA'05*, pages 305–322, 2005.

Table des figures

Résumé

L'objet de cette thèse est la diffusion numérique sécurisée réalisée à l'aide de courbes elliptiques.

Le premier chapitre est consacré au calcul de points de ℓ-torsion sur une courbe elliptique définie sur un corps fini de caractéristique p. Plus précisément, nous combinons un algorithme rapide de calcul d'isogénies dû à Bostan, Morain, Salvy et Schost avec l'approche p-adique suivie par Joux et Lercier. Nous obtenons ainsi le premier algorithme valide sans limitation sur ℓ et p dont la complexité est similaire à celle de l'algorithme proposé par Bostan *et al.*

Dans le deuxième chapitre, nous développons un modèle générique de groupes avec couplage qui généralise les modèles présentés auparavant dans la littérature. Nous fournissons un cadre général permettant de prouver dans ce modèle les hypothèses cryptographiques reliées au problème du logarithme discret sur des groupes avec couplage.

Dans le troisième chapitre, nous proposons et étudions un nouveau schéma de diffusion pour des récepteurs sans état. À la différence des schémas s'appuyant sur des techniques de recouvrement par des sous-ensembles définis par des arbres binaires, notre schéma considère que l'ensemble des récepteurs destinataires d'un message est décrit par des attributs. La taille du chiffré est linéaire en le nombre d'attributs utilisés dans cette description, mais ne dépend pas du nombre de destinataires. Par rapport à d'autres schémas basés sur des attributs, le déchiffrement nécessite des capacités de calculs bien plus faibles.

Le dernier chapitre est consacré à un schéma de chiffrement avec traçage de traîtres, c'est-à-dire conçu pour lutter contre le piratage dans la distribution sécurisée de contenus vers de nombreux destinataires. Nous proposons un nouveau schéma, utilisant des techniques de marquage de contenu, présentant un taux de chiffrement constant et une sécurité contre des décodeurs pirates puissants. Une particularité de ce schéma est la possibilité pour un destinataire de déchiffrer à la volée le contenu transmis.

Abstract

This thesis deals with secure digital broadcast designed with elliptic curves.

The first chapter is devoted to the computation of ℓ-torsion points on elliptic curves defined over a finite field of characteristic p. More precisely, we combine a fast algorithm for computing isogenies due to Bostan, Morain, Salvy and Schost with the p-adic approach used by Joux and Lercier. We obtain the first algorithm valid without any limitation on ℓ and p, the complexity of which is similar to the one of Bostan *et al.*'s algorithm.

In the second chapter, we develop a generic group model with pairing which generalizes the models seen so far in the literature. We provide then a general framework: cryptographic assumptions related to the discrete logarithm problem can be proved easily in this model.

In the third chapter, we describe and study a new broadcast encryption scheme for stateless receivers. The main difference between our scheme and the usual ones derived from the subset-cover paradigm over binary trees is that the group of selected receivers is described by attributes. The size of a ciphertext is linear in the number of attributes used in this description, but does not depend on the number of selected receivers. In comparison with previous attribute-based schemes, the decryption algorithm requires less computational resources.

The last chapter is devoted to a traitor tracing scheme designed to fight piracy when distributing data securely to multiple authorized receivers. We propose a new scheme, using watermarking techniques, with a constant transmission rate and security against powerful pirate decoders. In this scheme, a receiver can moreover decrypt the ciphertexts on the fly.